参考資料

User's Guide

STM8 から MSPM0 への移行ガイド

TEXAS INSTRUMENTS

概要

このアプリケーション ノートは、STMicroelectronics STM8L および STM8S プラットフォームから テキサス・インスツルメンツの MSPM0L および MSPM0C MCU エコシステムへの移行を支援します。このドキュメントでは、MSPM0 の開発およびツール エコシステム、コア アーキテクチャ、ペリフェラルに関する考慮事項、およびソフトウェア開発キットについて説明します。この目的は、2 つのファミリの違いを強調し、STM8 の開発環境に関する既存の知識を活用して、MSPM0 シリーズの MCU で迅速に開発を開始することです。

目次

図の一覧

表の一覧

商標

1 MSPM0 製品ラインアップの概要

1.1 はじめに

MSPM0 マイクロコントローラ (MCU) 製品は、強化された ARM® Cortex®-M0+ 32 ビット コア プラットフォームをベースとする MSP の高集積超低消費電力 32 ビット MCU ファミリの一部です。コスト最適化されたこれらの MCU は、高性能アナログ ペリフェラルの統合、拡張温度範囲のサポート、および小型フットプリント パッケージを実現します。TI の MSPM0 低消費電力 MCU ファミリは、アナログおよびデジタル回路をさまざまなレベルで内蔵したデバイスで構成されているため、エンジニアはプロジェクトの要件を満たす MCU を見つけることができます。MSPM0 MCU ファミリは、Arm Cortex-M0+ プラットフォームと超低消費電力のシステム アーキテクチャを組み合わせたもので、システム設計者は性能向上と消費電力低減を同時に実現できます。

MSPM0 MCU は、STM8 MCU に代わる競争力のある選択肢を提供します。このアプリケーション ノートは、デバイスの機能とエコシステムを比較することで、STM8 MCU から MSPM0 MCU への移行を支援します。

1.2 STM8 MCU と MSPM0 MCU の製品ラインアップの比較

表 1-1. TI MSPM0Lx、MSPM0Cx と STM8Lx、STM8Sx の比較

		ST マイクロ STM8 L シリーズ	ST マイクロ STM8 S シリーズ	TI MSPM0 Lx シリーズ	TI MSPM0 Cx シリーズ	TI MSPM0 Hx シリーズ
コア		STM8 CPU コア		ARM Cortex-M0+		
最大周波数		16MHz	24MHz	32MHz	24MHz、32MHz(1)	32MHz
電源電圧		1.6～3.6V	2.95～5.5V	1.62～3.6V	1.62～3.6V	4.5～5.5 V(2)
最高温度		-40～125 ℃	-40～125 ℃	-40～125 ℃	-40～125 ℃	-40～125 ℃
メモリ		64KB～2KB	128KB～4KB	64KB～8KB	16KB～8KB	64KB～32KB
RAM		最大 4 KB	最大 6 KB	最大 4 KB	最大 8 KB	最大 8 KB
GPIO (最大)		41	38	28	45	45
RTC		あり	いいえ	あり	あり (C1103 および C1104 は非対応)	あり
アナログ	ADC	最大 12 ビット x 28 チャンネル	最大 10 ビット x 16 チャンネル	1x 1.68Msps 12 ビット ADC (16 チャンネル)	1x 1.5Msps 12 ビット ADC (最大 27 チャンネル)	1x 1.5Msps 12 ビット ADC (最大 27 チャンネル)
	DAC	最大 12 ビット x 2 チャンネル	なし	8 ビット	8 ビット (C1103 および C1104 は非対応)	なし
	コンパレータ	3µs 伝搬遅延	なし	1x 高速	1x 高速 (C1103 および C1104 は非対応)	なし
通信	UART	1 Mbit/s、最大 3 UART	1 Mbit/s、最大 2 UART	2x 4 Mbit/s	2x 4 Mbit/s	3x 4 Mbit/s
	I2C	1、100、400 Kbit/s	1、100、400 Kbit/s	2、最大 1 Mbit/s	1、最大 1 Mbit/s	2、最大 1 Mbit/s
	SPI	10 Mbit/s	10 Mbit/s	16 Mbit/s	12 Mbit/s	16 Mbit/s
	CAN	なし	1Mbit/s、最大 3 メールボックス	なし	なし	なし
	LIN	UART のサポート		UART のサポート		
その他の主なペリフェラル/特長		LCD ドライバ ビープ音 タッチセンシング (STM8L CT ライブラリ) DMA IR インターフェース	ビープ音 タッチセンシング (STM8S RC ライブラリ)	2x オペアンプ LCD (L2228) DMA	最小サイズの QFN パッケージ (2 x 2) および BGA パッケージ (0.861 x 1.6)、0.5/0.65mm ピッチ パッケージ、業界とのピン互換 DMA	5V 電源、DMA、ウィンドウウォッチドッグ タイマ
タイマ番号		3、4、5	3/4	7	5 (C1103 および C1104 は 3 つのタイマをサポート)	4

表 1-1. TI MSPM0Lx、MSPM0Cx と STM8Lx、STM8Sx の比較 (続き)

	ST マイクロ STM8 L シリーズ	ST マイクロ STM8 S シリーズ	TI MSPM0 Lx シリーズ	TI MSPM0 Cx シリーズ	TI MSPM0 Hx シリーズ
ピン数	最大 68	最大 68	16～80 ピン	8～48 ピン	20～48 ピン
低消費電力	アクティブ:16MHz 動作時 1.6mA ホールト:0.3μA	アクティブ:16MHz 動作時 1.8mA ホールト:5μA	アクティブ: 71μA/MHz スタンバイ: 1μA	アクティブ: 100μA/MHz スタンバイ: 5μA	該当なし

(1) MSPM0C1103 および MSPM0C1104:24MHz、MSPM0C1105 および MSPM0C1106:32MHz
(2) H シリーズの最初のデバイの対応電圧は 4.5～5.5V で、将来的により幅広い電源電圧にも対応する予定です。

2 エコシステムと移行

MSPM0 MCU は、ハードウェアおよびソフトウェアの大規模なエコシステムによってサポートされており、リファレンス デザインやコード サンプルを利用して設計をすぐに開始できます。MSPM0 MCU は、オンライン リソース、MSP Academy を使用したトレーニング、TI E2E™ サポート フォーラムによるオンライン サポートによってもサポートされています。

2.1 エコシステムの比較

表 2-1. エコシステムの比較

機能	STM8 デバイス	MSPM0 デバイス
コード ソース	標準的なペリフェラル ライブラリ (組み込みソフトウェア ドライバおよびサンプルをまとめたもの) のファームウェア パッケージ	MSPM0-SDK (DriverLib、ミドルウェア、RTOS、サンプル コード)
IDE	STVD、IDEA、IAR-EWSTM8、iSYS-WinIDEA、Arduino IDE	CCS、IAR、Keil
ソフトウェアの設定	STM8CubeMX	SysConfig
フラッシュ プログラミング ツール	FLASHER-STM8 STVP(STM8)	UniFlash
プログラマ	FlashRunner	MSP-GANG、C-GANG
デバッガ	ST-Link v2、ic5000	XDS110、J-LINK
ハードウェア	STM8-SO8-DISCO、STM8S-DISCOVERY、STM8SVLDISCOVERY、STM8L-DISCOVERY、NUCLEO-8S208RB、NUCLEO-8L152R8	LP-MSPM0G3507 LaunchPad、LP-MSPM0L1306 LaunchPad、LP-MSPM0C1104 LaunchPad、LP-MSPM0C1106 LaunchPad、LP-MSPM0H3216 LaunchPad

MSPM0 エコシステムの概要を、図 2-1 に示します。

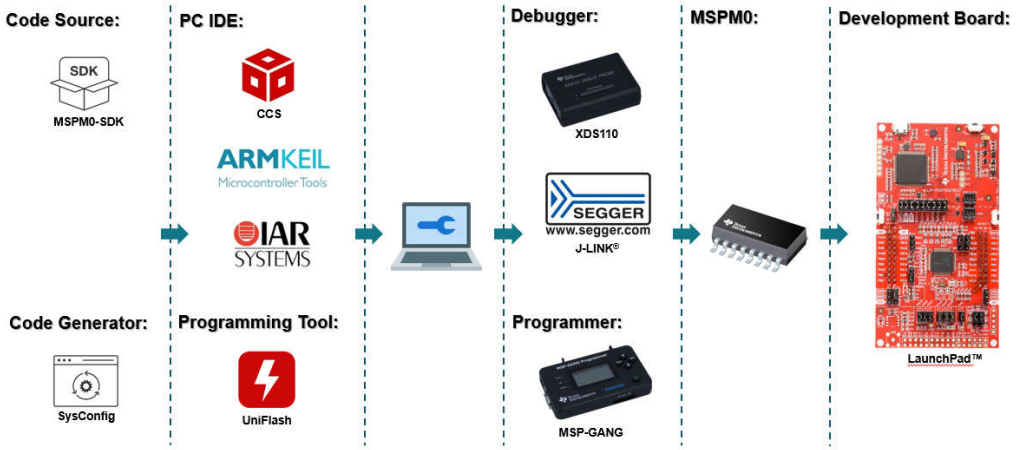


図 2-1. MSPM0 エコシステムの概要

2.1.1 MSPM0 ソフトウェア開発キット (MSPM0 SDK)

MSPM0 SDK には豊富なサンプル コードが含まれており、エンジニアがテキサス・インスツルメンツの MSPM0+ マイコンでアプリケーション開発を行いやすくなっています。各機能分野とすべてのサポート デバイスの使用法を解説しており、設計を開始する際の出発点になります。図 2-2 に、MSPM0 SDK の構造を示します。

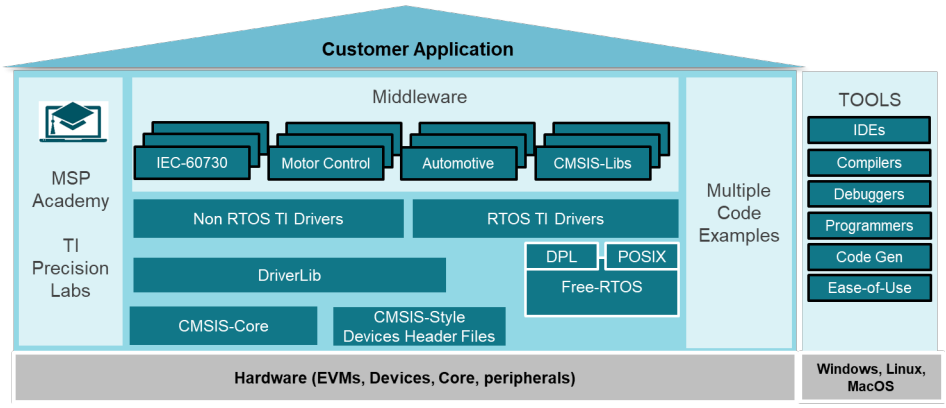


図 2-2. MSPM0 SDK の構造

MSPM0 SDK は、MSPM0-SDK サポートソフトウェア | TI.com からダウンロードできます。MSPM0 SDK には以下の 4 つのフォルダが含まれています:

例: サンプル フォルダは RTOS と非 RTOS のサブフォルダに分割されています (現在、非 RTOS のみがサポートされています)。これらのフォルダには各 LaunchPad™ のサンプルが含まれており、下位レベルの Driverlib サンプル、上位レベルの *TI ドライバ* サンプル、GUI Composer、LIN、IQMath などの*ミドルウェア*のサンプルの機能に基づいて編成されています。

Docs: ユーザーガイドや API ガイドなど、関連するすべての資料が付属しています。

出典: すべてのドライバとミドルウェアを対象とするソース コードとライブラリ。

Tools: MSPM0 アプリケーションの開発およびテストに役立つツールのセット。

STM8 については、ST が無償の標準ペリフェラルライブラリの他、RTOS、ブートローダーなどを提供しています。また、STM8 にはタッチセンサー、モーター制御、LIN ライブラリなど多くのミドルウェアや応用分野があり、MSPM0 もほぼ同様に対応しています。

ほとんどの MSPM0 のサンプルは SysConfig をサポートしており、デバイスの構成を簡素化し、ソフトウェアを迅速に開発できるようにします。

その他の参考ドキュメントを以下に示します:

- MSPM0 SDK ユーザー ガイド
- MSPM0 ツール ガイド
- Driverlib API ガイド

2.1.2 MSPM0 による IDE サポート

統合開発環境 (IDE) とは、プログラミングを効率よく行うためのソフトウェアで、通常はエディタ、コンパイラ、デバッガなどを備えています。

STM8 の代表的な IDE は STMicroelectronics が提供する STVD です。STVD ではサンプルコードのダウンロードが可能で、使いやすい Eclipse ベースのコードエディタも備えています。ただし、STVD にはアセンブリ用のコンパイラしかなく、C 言語で開発するには別途「Cosmic ツール」と呼ばれる C コンパイラをインストールする必要があります。Cosmic は STM8 用の C コンパイラを提供しています。最大 32KB までのコードであれば無料で利用可能です。そのため、多くの STM8 ユーザーが IAR を使用してプロジェクトを開発しており、これは MSPM0 でも同様に使用可能です。

TI では、TI のマイコン (MCU) と組込みプロセッサの製品ラインアップに対応する Code Composer Studio™ IDE (CCS) の使用を強くお勧めします。具体的に、CCS は、最適化された C/C++ コンパイラ、ソース コード エディタ、プロジェクトのビルド環境、デバッガ、プロファイラ、その他の多くの機能を含む、組込みアプリケーションの開発とデバッグに使用する一連のツールで構成されています。CCS は完全に無料で使用でき、2 つの形式で提供されています。

これら 2 つの IDE の違いと類似点を表 2-2 に示します。

表 2-2. CCS と STVD の比較

複数の IDE	CCS	STVD
ライセンス	未使用	未使用
コンパイラ	TI Arm Clang、GCC	Cosmic、raisonance
電流消費量の確認機能が IDE に統合	EnergyTrace	サポートなし (STM8CubeMX でサポート)
周辺機能の API 関数サポート	非対応	非対応
表示言語	英語	英語
ファイル変換	16 進ファイル、バイナリファイル、Motorola S レコードファイル、Ti_txt ファイル	16 進ファイル、バイナリファイル、Motorola S レコードファイル
コード生成用 GUI	SysConfig	STM8CubeMX

CCS は、統合型の TI Resource Explorer に、MSPM0 デバイス構成と SysConfig からの自動コード生成、MSPM0 サンプル コードとアカデミー トレーニングを統合しています。さらに、CCS は、一体型の開発ツールを提供します。

CCS に加えて、MSPM0 デバイスは、表 2-3 に示す業界標準 IDE でもサポートされています。

- CCS：https://www.ti.com/tool/CCSTUDIO
- IAR：https://www.iar.com/
- Keil：https://www.keil.com/

表 2-3. MSPM0 でサポートされている IDE の概要

複数の IDE	CCS	IAR	Keil
ライセンス	未使用	有料	有料
コンパイラ	TI Arm Clang、GCC	Arm 用 IAR C/C++ コンパイラ	ARM コンパイラ バージョン 6
ディスクサイズ	0.88G (ccs20.1.1)	6.33G (Arm 8.50.4)	2.5G (µVision® V5.37.0)
XDS110	対応	対応	対応
J-Link	対応	対応	対応
EnergyTrace	対応	いいえ	いいえ
MISRA-C	いいえ	対応	いいえ
セキュリティ	いいえ	対応	いいえ
ULINKplus	いいえ	いいえ	対応
機能安全	いいえ	対応	対応

CCS の使用方法と機能の一部をセクション 2.2.2.2 に示します。その他の参考資料を以下に示します：

- CCS クイック スタート ガイド
- CCS
- CCS トレーニング動画
- CCS ユーザーガイド
- IAR クイック スタート ガイド
- IAR トレーニング動画
- IAR ユーザーガイド
- Keil クイック スタート ガイド
- Keil トレーニング動画
- Keil を始める

2.1.3 SysConfig

STM8CubeMX と同様に、SysConfig は、ピン、ペリフェラル、無線、サブシステム、他の機能を構成するために使用できる、直観的で包括的なグラフィカル ユーティリティのコレクションです (図 2-3 参照)。SysConfig を使用すると、コンフリクトの管理、表面化、解決をビジュアルな方法で実行できるので、より多くの時間をアプリケーションの差異化に割り当てることができます。このツールの出力には C ヘッダとコード ファイルが含まれており、MSPM0 SDK サンプルと組み合わせて使用することも、カスタム ソフトウェアの構成に使用することもできます。SysConfig は CCS に統合されていますが、Keil および IAR でも使用できます。

SysConfig は、次の URL からダウンロードできます:SYSCONFIG IDE、構成、コンパイラまたはデバッガ | TI.com。

SysConfig は、IDE なしで実行できます。スタンドアロン バージョンは、コード生成やデバイスの機能評価には使用できますが、サンプルを実行することはできません。

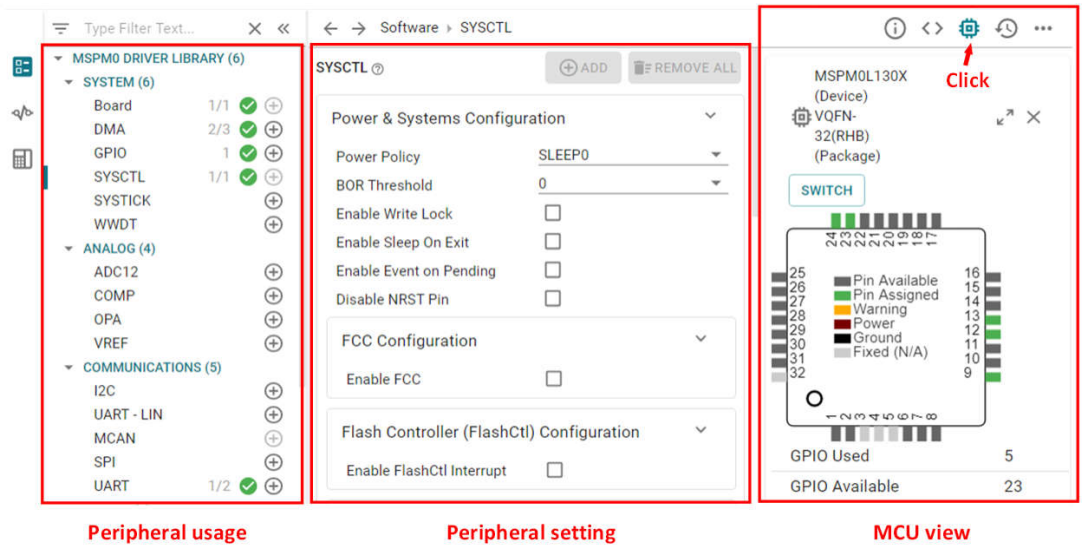


図 2-3. MSPM0 SysConfig

ここでは、SysConfig と STM8CubeMX の共通機能と異なる機能を紹介します:

- どちらも、作成、保存、以前に保存したプロジェクトのインポートを行うことができます。STM8CubeMX のプロジェクトは .ioc8 ファイルとして保存され、他の .ioc8 ファイルと一緒に任意の場所に保存できます。ただし、STM8CubeMX は C コード生成をサポートしていません。そのため、SysConfig と比べて開発時間が長くなる場合があります。
- どちらも、チップ ビューから確認できるピンアウトとクロック ツリーの設定を簡単に行うことができます。
- STM8CubeMX では消費電力の計算ができますが、MSPM0 では同様のことが CCS IDE 上で行えます。
- SysConfig は、より包括的な設定に対応しています。周辺機器のピンアウトまたは単純な設定のみを設定できる STM8CubeMX とは異なり、SysConfig ではより具体的で詳細な初期化設定を行うことができます。図 2-4 に示すように、SysConfig では、GPIO のタイプのほか、内部プルアップ抵抗やプルダウン抵抗などの機能の設定が可能です。
- STM8CubeMX にはアップデート機能が搭載されており、自動または手動で更新の有無を確認するように設定でき、STM8CubeMX の自己アップデートに対応しています。SysConfig にはその機能はなく、最新バージョンを SysConfig ダウンロードからダウンロードする必要があります。

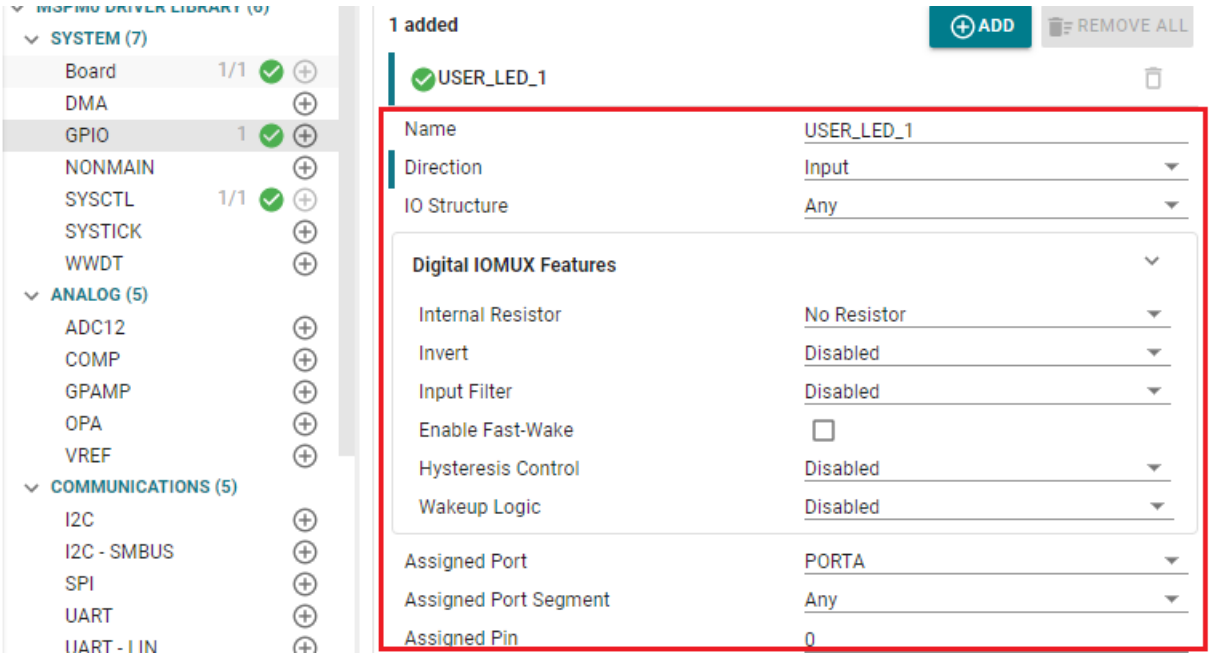


図 2-4. SysConfig 内の GPIO の設定

2.1.4 デバッグ ツール

STM は、単線インターフェースモジュール (SWIM) およびデバッグモジュール (DM) をサポートしています。回路内デバッグモードまたは回路内プログラミングモードは、超高速メモリプログラミングを特徴とする単線式ハードウェアインターフェイスを介して制御されます。また、回路内デバッグモジュールと組み合わせることで、非割り込みエミュレーションモードも提供され、これにより回路内デバッガが強化され、フル機能エミュレータと同等の性能を発揮します。STM8 の一般的なデバッガは ST-LINK です。SWIM および JTAG、シリアルワイヤーデバッグ (SWD) インターフェイスは、アプリケーションボードに配置されている任意の STM8 マイコンとの通信に使用します。

MSPM0 の場合、デバッグ サブシステム (DEBUGSS) は、シリアル ワイヤ デバッグ (SWD) の 2 線式物理インターフェイスを、デバイス内の複数のデバッグ機能に接続します。MSPM0 デバイスは、プロセッサの実行、デバイスの状態、電力状態 (EnergyTrace テクノロジーを使用) のデバッグをサポートしています。デバッガの接続の詳細については、図 2-5 を参照してください。

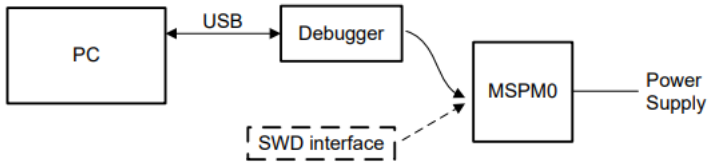


図 2-5. MSPM0 のデバッグ

MSPM0 は、標準的なシリアル ワイヤ デバッグ用の XDS110 および J-Link デバッガをサポートしています。

テキサス・インスツルメンツの XDS110 は、TI の組込みプロセッサ用です。XDS110 は、TI 20 ピン コネクタ (TI 14 ピンおよび Arm 10 ピンおよび 20 ピンコネクタ用の変換アダプタも利用可能) を使用してターゲット ボードに接続し、USB 2.0 ハイスピード (480Mbps) でホスト PC に接続します。XDS110 は、単一のユニットで幅広い規格 (IEEE1149.1、IEEE1149.7、SWD) をサポートしています。すべての XDS デバッグ プローブは、ETB (Embedded Trace Buffer、組込みトレース バッファ) 搭載のすべての Arm と DSP プロセッサに対し、コア トレースとシステム トレースをサポートしています。詳細については、XDS110 デバッグ プローブを参照してください。

J-Link デバッグ プローブは、デバッグとフラッシュ プログラミングの経験を最適化するための最も一般的な選択肢です。記録的なブレークダウンを実現するフラッシュ ローダ、最大 3MiB/s の RAM ダウンロード速度、MCU のフラッシュ メモリ内に無制限のブレークポイントを設定する機能を活用できます。また、J-Link は CortexM0+ を含む幅広い CPU とアーキテクチャもサポートしています。詳細については、J-Link デバッグ プローブのページを参照してください。

表 2-4 に、MSPM0 をサポートする XDS110 と J-LINK デバッガの機能の違いの概要を示します。

表 2-4. MSPM0 デバッガの比較

特長	XDS110	XDS110 OB (1)	J-Link
cJTAG (SBW)	✓	✓	✓
BSL (2) ツール	✓	✓	
バックチャネル UART	✓	✓	2.5G (µVision V5.37.0)
電源	1.8 ～ 3.6 V	3.3、5V	5V
IDE (3):CCS	✓	✓	✓
IDE:サード パーティー (4)	IAR、Keil	IAR、Keil	IAR、Keil

(1) XDS110 OB は、オンボードの XDS110 を意味します。
(2) BSL はブート ストラップ ローダを意味します。
(3) IDE は、統合開発環境のことです。
(4) サード パーティーには IAR と Keil が含まれています。

2.1.5 LaunchPad

STM8 と同様に、MSPM0 にも迅速な開発を支援するための対応する LaunchPad 開発キットが用意されています。

LaunchPad キットは使いやすい評価基板で、MSPM0 の開発を開始するために必要なものがすべて含まれています。これには、EnergyTrace™ テクノロジーを使用したプログラミング、デバッグ、消費電力測定用のオンボード デバッグ プローブが含まれています。MSPM0 LaunchPad には、オンボード ボタン、LED、温度センサなどの回路も搭載されています。さまざまなブースタパック プラグイン モジュールをサポートする 40 ピンのブースターパック™ プラグイン モジュール ヘッダーにより、迅速で簡単なプロトタイプ製作が可能になります。ワイヤレス接続、グラフィカル ディスプレイ、環境センシングなどの機能も迅速に追加できます。

図 2-6 に LaunchPad (MCU と XDS110 デバッガを含む) の概要を示します。ジャンパーを取り外せば、J-Link など他のデバッガを使用して MCU のデバッグを行うこともできます。

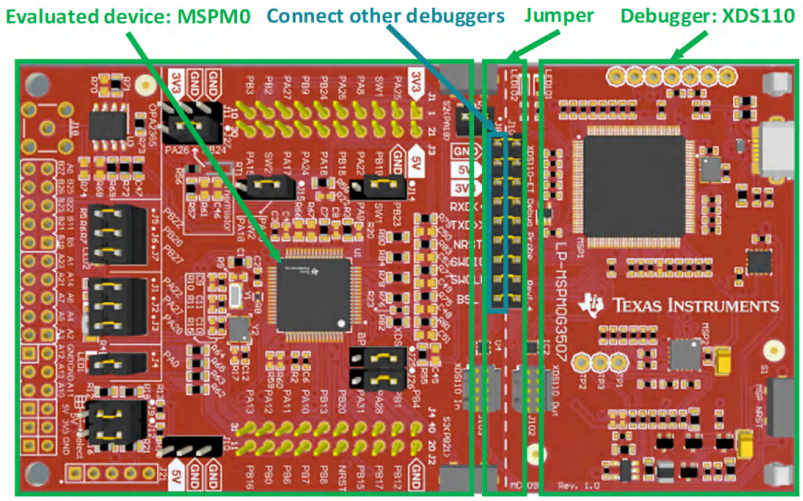


図 2-6. MSPM0G3507 Launchpad の概要

ジャンパ絶縁端子台には、電源 (GND、5V、3.3V)、UART (RXD、TXD)、リセットピン、ARM デバッグチャネル (SWDIO、SWCLK)、BSL が含まれています。

ジャンパ キャップに加えて、Launchpad の右側にある標準的な Arm Cortex 10 ピンコネクタ (図 2-7 を参照) を使用して書き込みを行うこともできます。Cortex デバッグ コネクタは、JTAG デバッグ、シリアル ワイヤ デバッグ、シリアル ワイヤ ビューア (シリアル ワイヤ デバッグ モードを使用している場合は SWO 接続経由) 機能をサポートしています。

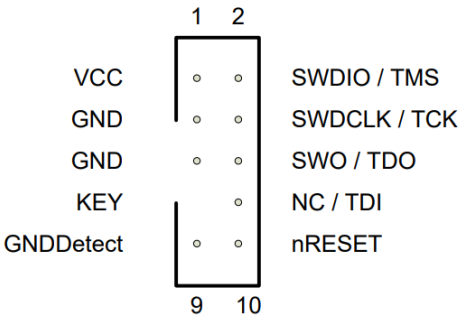


図 2-7. Arm Cortex 10 ピンの定義

図 2-8 に、MSPM0G3507 LaunchPad の機能の一部を示します。

LaunchPad の下部には BoosterPack 用のコネクタがあり、専用の機能モジュールを直接差し込んで、すばやくプロトタイプを構築できます。また、DuPont ワイヤーを単独で引き出して素早く使用することも可能です。LaunchPad には両側にユーザー定義ボタン、温度センサー、光センサー、単色 LED、その下に RGB LED が搭載されています。

- LP-MSPM0G3507 LaunchPad 開発キット LP-MSPM0G3507 評価ボード | TI.com
- LP-MSPM0L1306 LaunchPad 開発キット LP-MSPM0L1306 評価ボード | TI.com
- LP-MSPM0C1104 LaunchPad 開発キット LP-MSPM0C1106 評価ボード | TI.com

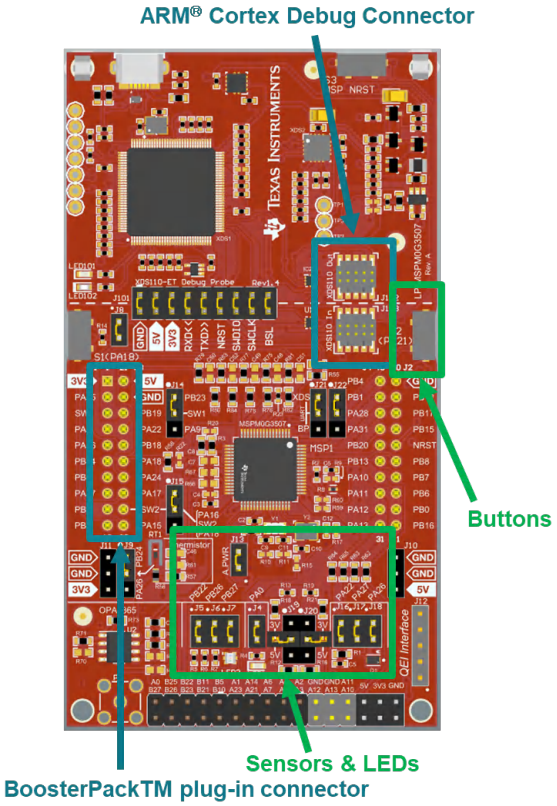


図 2-8. MSPM0G3507 Launchpad の特徴と機能

2.2 移行プロセス

MSPM0 へのスムーズな移行のために、図 2-9 に詳細な手順をフロー形式で示します。各ステップについて詳しく説明されており、具体例も続くセクションで紹介されています。

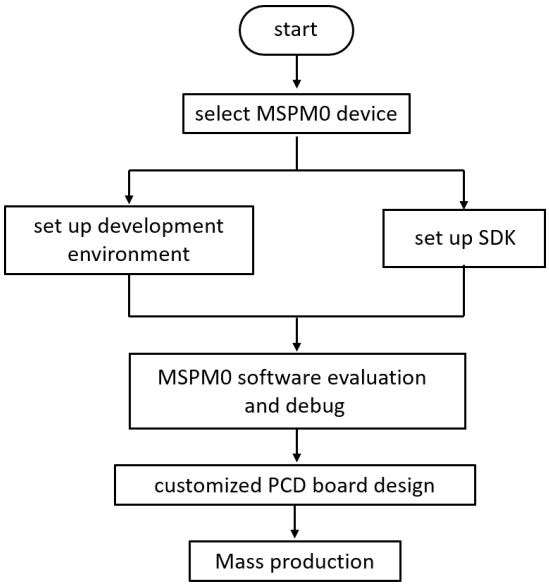


図 2-9. MSPM0 の移行フローチャート

2.2.1 ステップ 1 : 適切な MSPM0 MCU を選択する

移行の最初のステップは、アプリケーションに適した MSPM0 デバイスを選択することです。図 2-10 に MSPM0 C、L、G シリーズファミリの製品ラインアップを示します。これらは、TI の公式 Web サイトで参照できます。どちらの製品ラインアップも、メモリ容量やパッケージに基づいてデバイスを分類しているため、目的に合ったデバイスを選びやすくなっています。

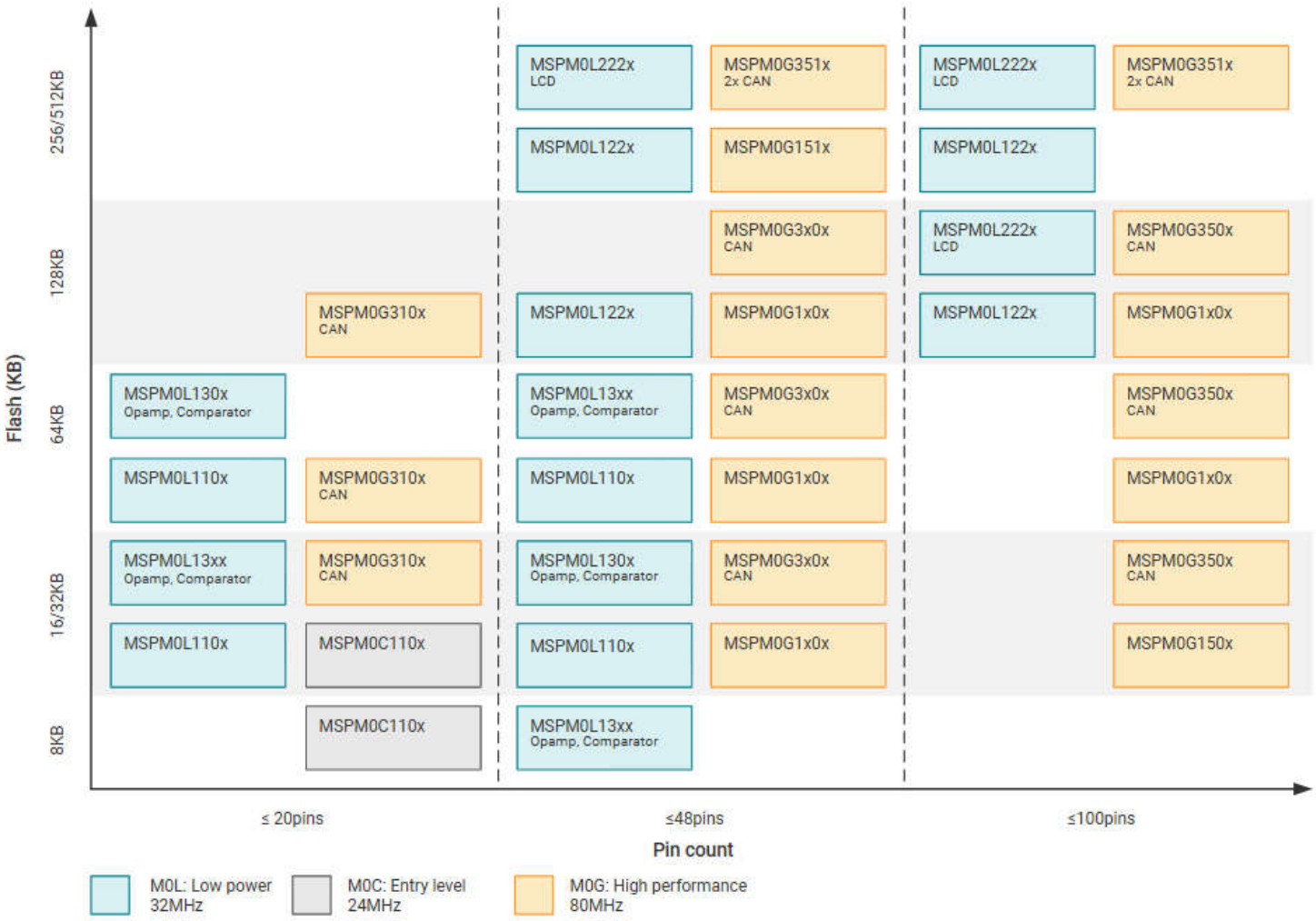


図 2-10. MSPM0C、MSPM0L、MSPM0G シリーズの製品ラインアップ

特定のデバイスを絞り込むには、製品選定ツールが重要な役割を果たします。このリンクでは、左のフィルタを使用して、マイコンの周辺機器の要件に基づいて初期のスクリーニングを実行します。たとえば、UART の番号に一致する MCU を絞り込む場合、フィルタ ツールで直接条件を選択すると、右側に該当する MCU デバイスが表示されます (図 2-11 を参照)。また、左上の検索ボックスからデバイス名を入力することで、該当デバイスの詳細情報ページに直接アクセスすることもできます。

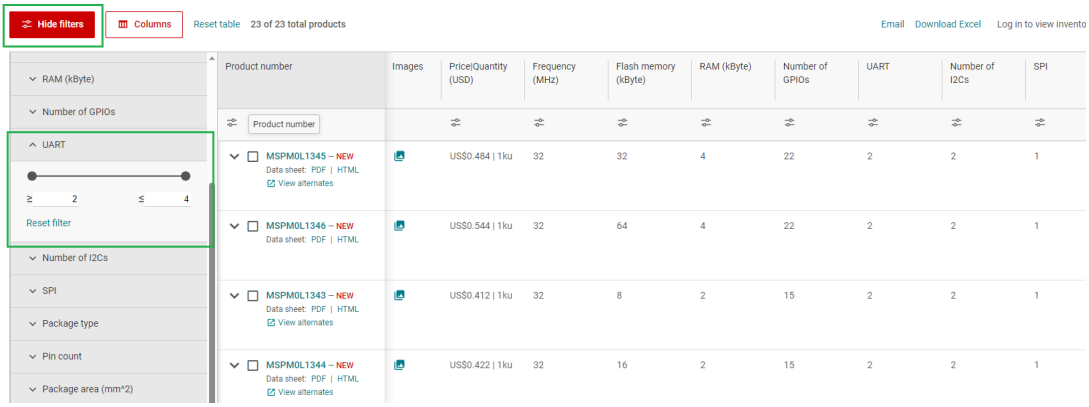


図 2-11. MSPM0 製品選択ツール

デバイスページでは、データシート、テクニカル リファレンス マニュアル (TRM)、正誤表などの主要ドキュメントの検索、ダウンロードを容易に行うことができます (図 2-12 を参照)。デバイス固有のデータシートでは、対象の MSPM0 の仕様や機能に関する情報が紹介されています。デバイス固有の TRM では、MSPM0 シリーズの使用方法や特徴が紹介されています。デバイス固有の正誤表では、MSPM0 関連のシリーズまたはバージョンの説明が紹介されています。

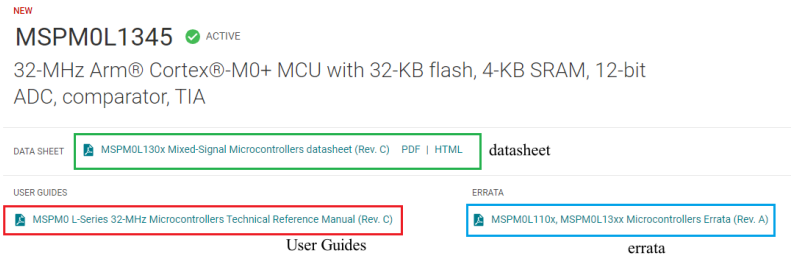


図 2-12. MSPM0 の重要ドキュメント一覧

図 2-13 に示すように、Web サイトには MSPM0 に関連する技術ドキュメントが掲載されており、なかでも代表的なものがアプリケーション ノートです。

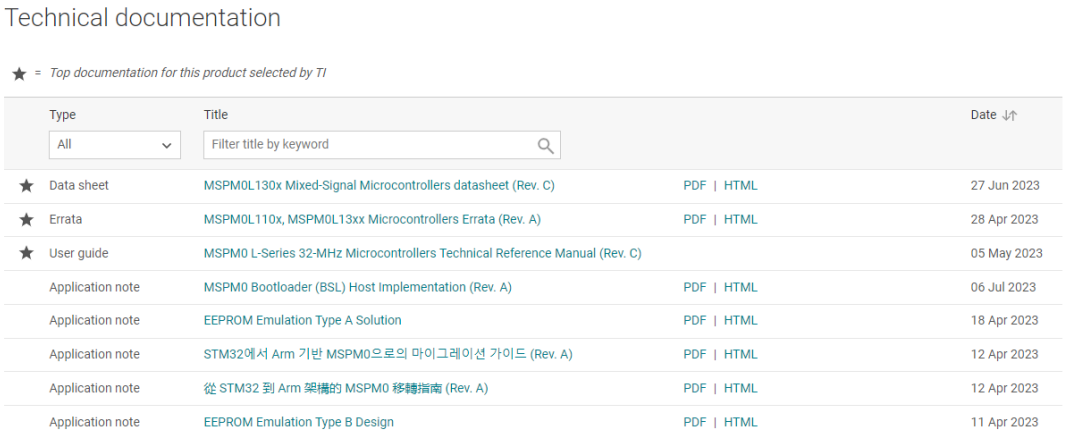


図 2-13. MSPM0 関連技術ドキュメント一覧

選択後、「注文と品質」セクションで価格やその他の情報を確認できます (図 2-14 を参照)。

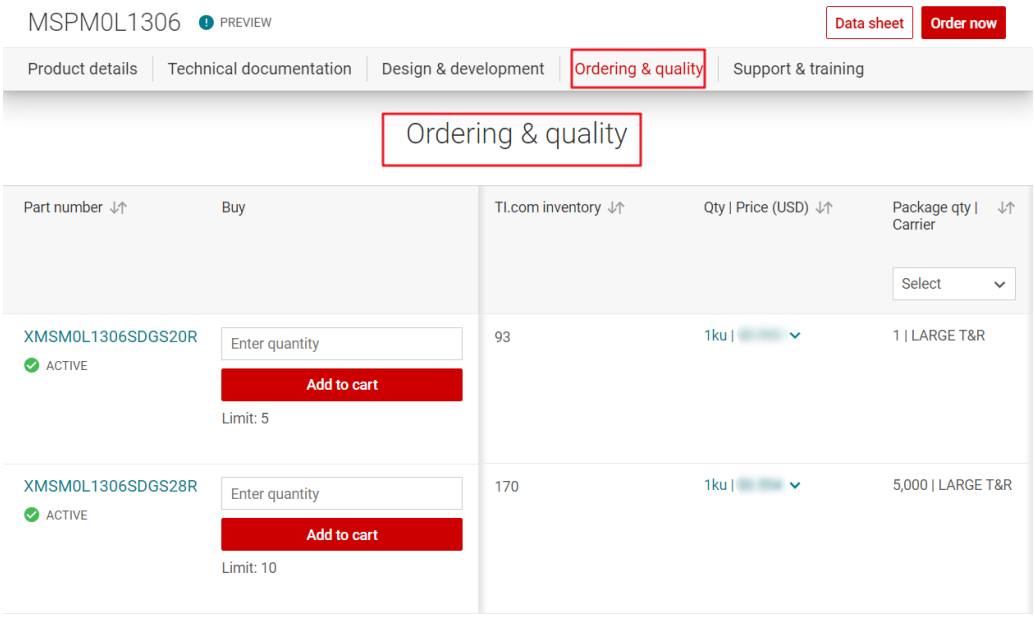


図 2-14. 注文と品質情報の表示画面

2.2.2 ステップ 2.IDE の設定と CCS の簡単な説明

2.2.2.1 IDE の設定

TI の CCS が選択された IDE です。

1. リンクをクリックしてダウンロードし、インストールを開始します。インストール プロセスで、[次へ] 押し続けます。

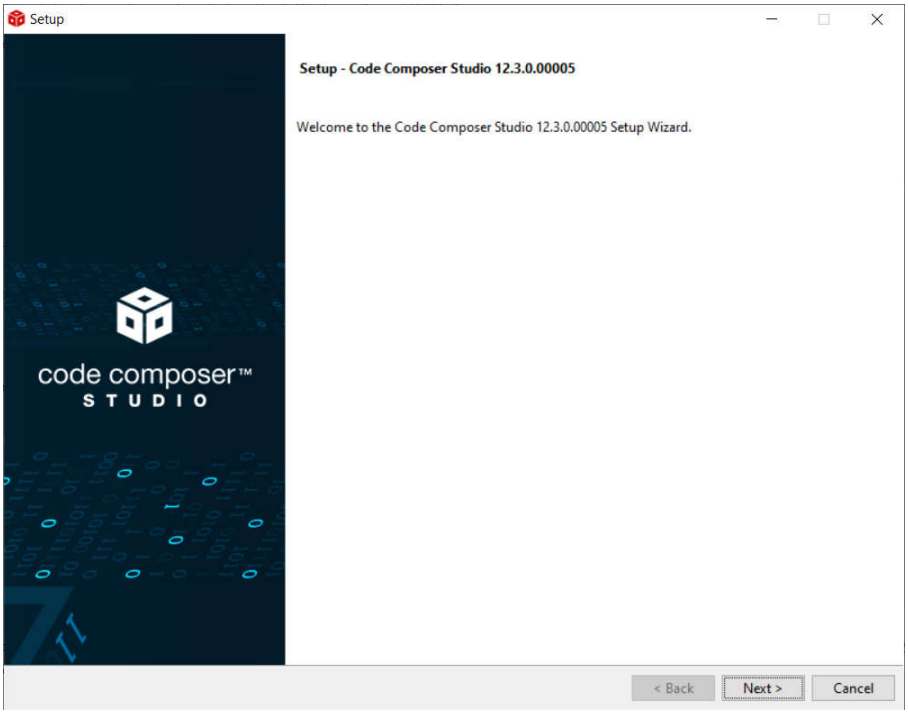


図 2-15. CCS のインストール

2. MSPM0 サポートコンポーネントを選択します。

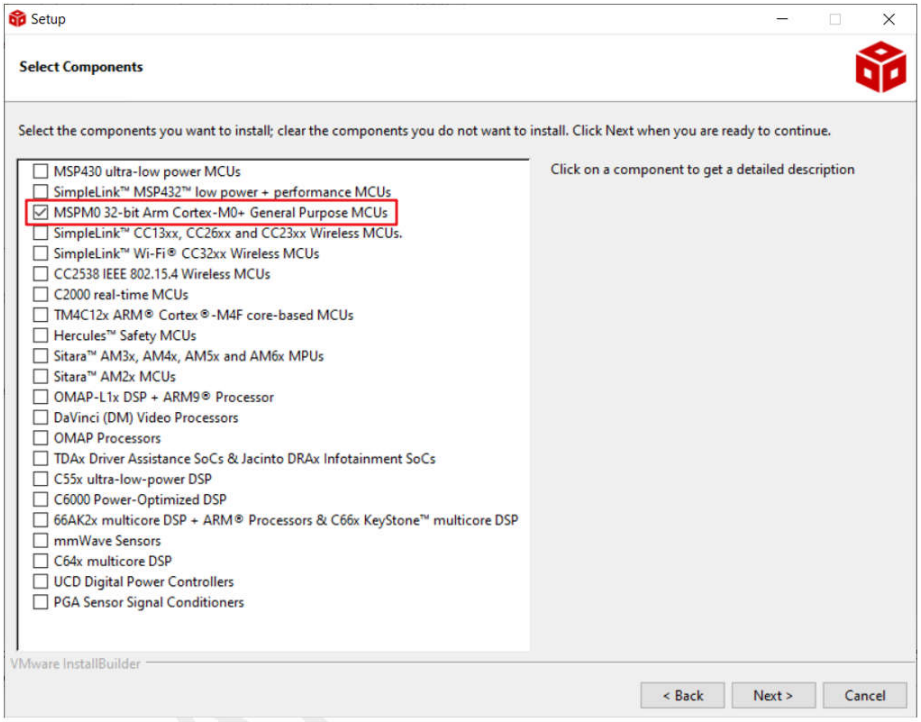


図 2-16. CCS のインストール - MSPM0 サポートの選択

3. 必要に応じて、J-link を選択します。

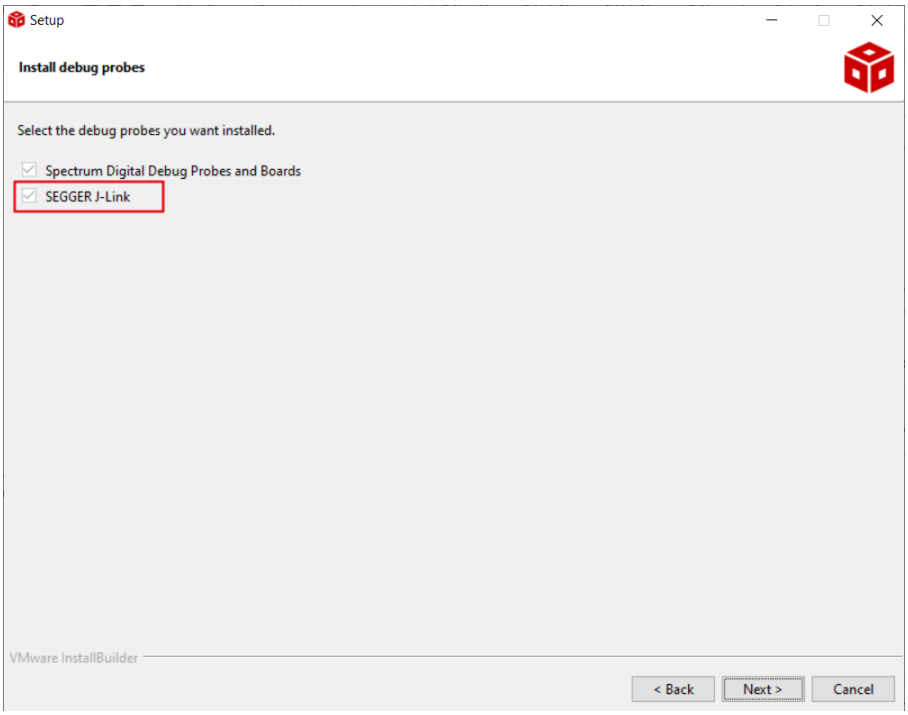


図 2-17. CCS のインストール - J-Link ダウンロードの選択

4. CCS のダウンロードを完了します。

2.2.2.2 CCS の簡単な説明

1. 新しいワークスペースを起動します。ワークスペースとは、インポートしたプロジェクトをコピーするアドレスです。

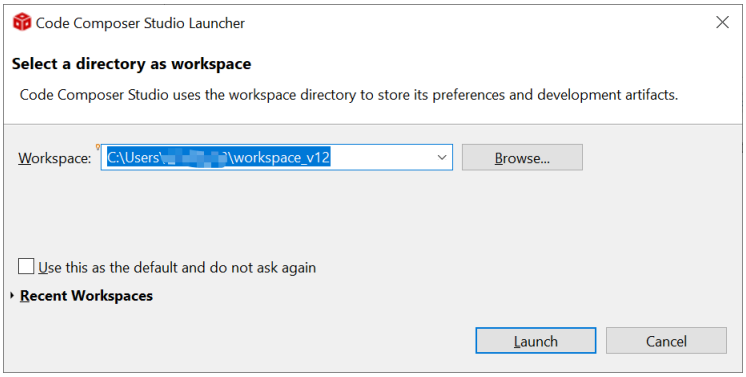


図 2-18. CCS Launch Workspace

2. 新しいプロジェクトを作成する場合は、[ファイル] → [新規作成] → [CCS プロジェクト] に移動します。ここで実施すべき重要な作業が 2 つあります。1 つ目は MSPM0 デバイスの選択、もう 1 つは接続の選択です (図 2-19 を参照)。選択した後、プロジェクト名を追加して [完了] ボタンを押すと、プログラムの作成が可能になります。TI では、CCS の使用方法を紹介するセクション 2.2.4 から MSPM0 SDK のサンプルを見つけることを推奨します。

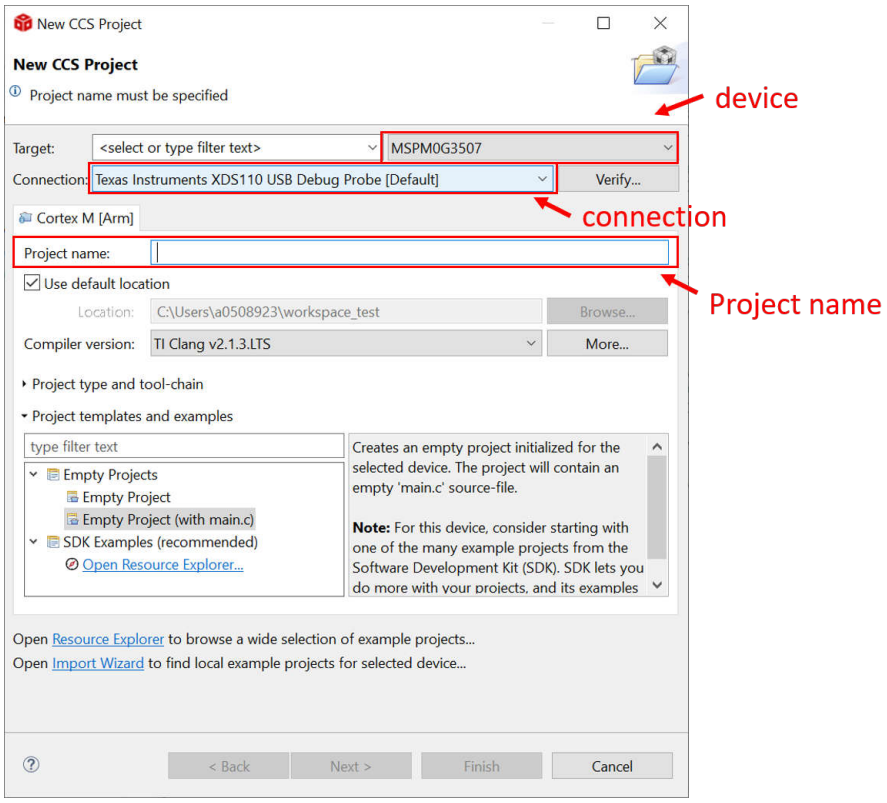


図 2-19. CCS で新しいプロジェクトを作成する

3. 図 2-20 および図 2-21 では、CCS 機能について簡単な説明をしています。

ショートカットキーの機能:

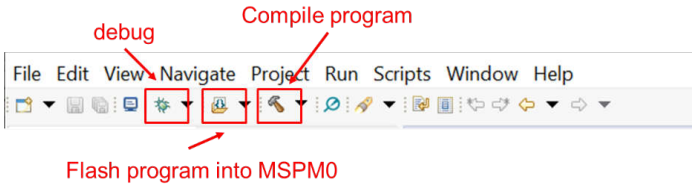


図 2-20. よく使う機能

デバッグ機能:

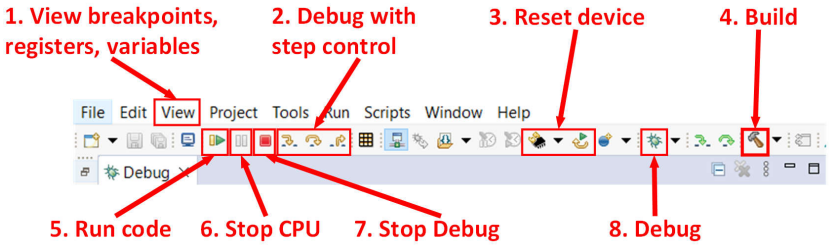


図 2-21. よく使うデバッグ機能

プロジェクトプロパティでよく使う設定：

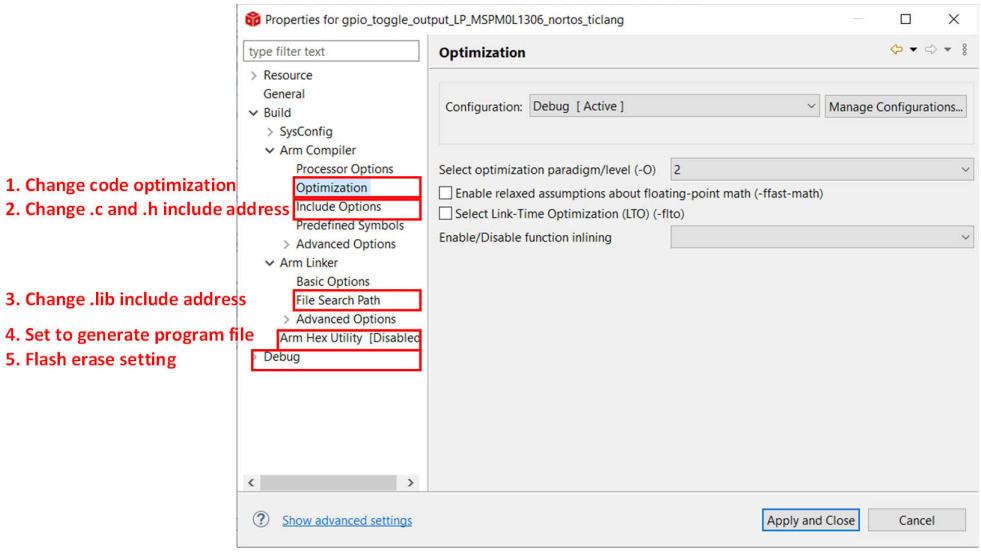


図 2-22. よく使うプロジェクト設定

詳細については、『MSPM0 MCU 用 Code Composer Studio IDE バージョン 12.4 以上』を参照してください。

2.2.3 ステップ 3.MSPM0 SDK の設定と MSPM0 SDK の簡単な説明

STM8 には「標準ペリフェラル ライブラリ」が提供されており、開発者はこれを使用して STM8 マイコンの機能を簡単に活用し、さまざまな用途に対応できます。TI も同様のサポートを提供しています。

2.2.3.1 MSPM0 SDK の設定

1. リンクをクリックして、MSPM0 SDK をダウンロードします。

図 2-23. MSPM0 SDK のダウンロード

2. [次へ] を押して SDK をインストールします。

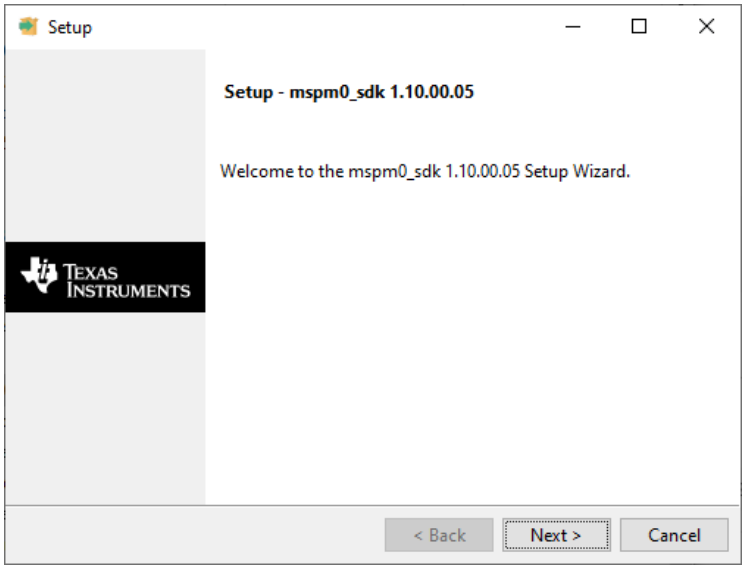


図 2-24. MSPM0 SDK のインストール

3. MSPM0 SDK のダウンロードを完了します。

2.2.3.2 SDK の簡単な説明

ダウンロードが完了すると、ファイルの内容が SDK フォルダ (デフォルトでは *C:/ti/mspm0_sdk_xxx*) に表示されます (図 2-25 を参照)。その中で、主に使用されるフォルダは、sample、docs フォルダです。

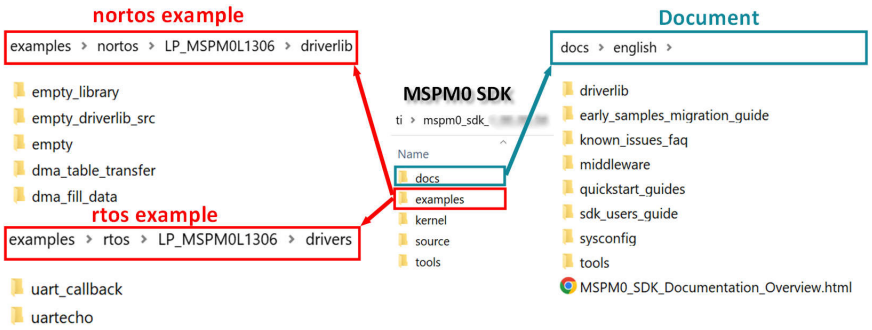


図 2-25. MSPM0 SDK フォルダ

表 2-5 に、サンプル カバレッジの概要を示します。

表 2-5. MSPM0 のカバレッジ例

SDK でサポート	プラットフォーム			
IDE	CCS		Keil	IAR
コンパイラ	TI Arm-Clang	GUN Arm	Arm/Keil コンパイラ	IAR Arm コンパイラ
RTOS	FreeRTOS			
サンプル コード	Driverlib、TI Dirvers (ドライバ)			

nortos の例には、ユーザーが独自のプロジェクトを作成できるように、空のプロジェクトが 3 つ用意されています。表 2-6 に、それぞれ違いを示します。

表 2-6. 空のプロジェクトの説明

例	SysConfig を使用します	プロジェクトにライブラリファイルを含めます
空	あり	いいえ
empty_library	いいえ	あり
empty_driverlib_src (推奨)	あり	あり

Docs フォルダの構造と重要なドキュメントを図 2-26 に示します。

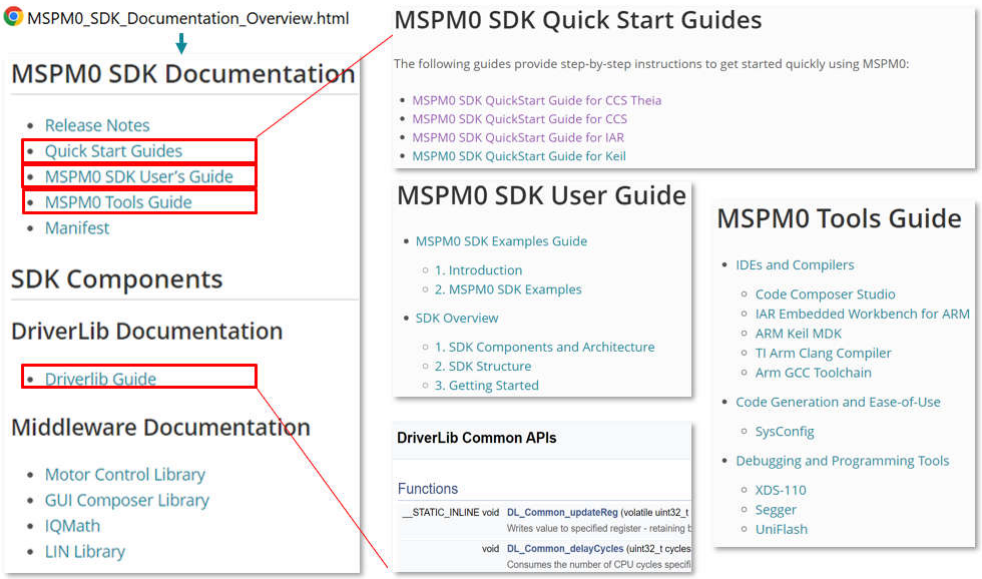


図 2-26. ドキュメントの概要

2.2.4 ステップ 4 : ソフトウェア評価

以下に、サンプルを CCS に移植する手順を示します。

1. メニューから*プロジェクト*を選択し、CCS プロジェクトをインポートします。

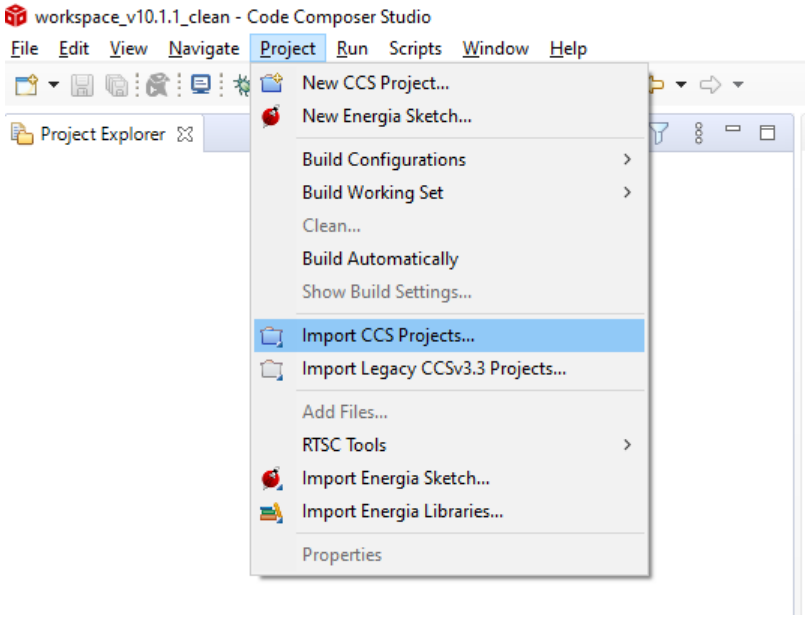


図 2-27. CCS プロジェクトのインポート

2. SDK からプログラムを選択します。MSPM0L1306 を例にします。

   \mspm0_sdk_1_10_00_05\examples\nortos\LP_MSPM0L1306\driverlib

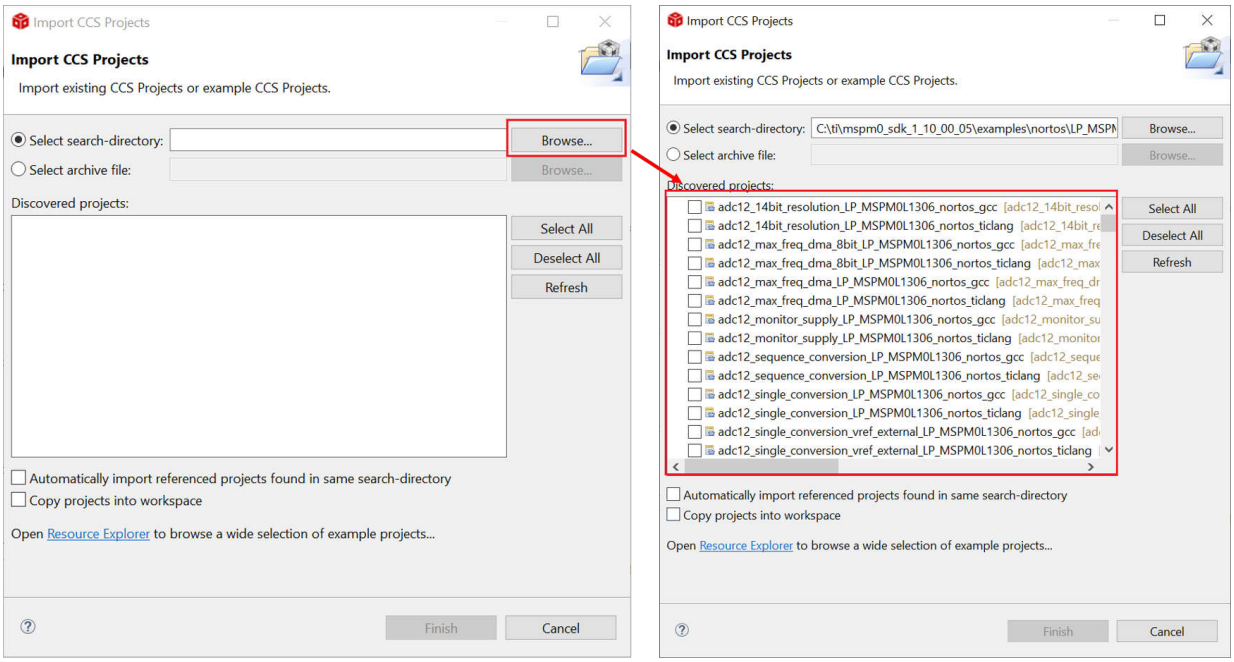


図 2-28. SDK からプログラムを選択します

ファイルをインポートできない場合、ワークスペースの下にある同じ名前のプロジェクトを削除します。

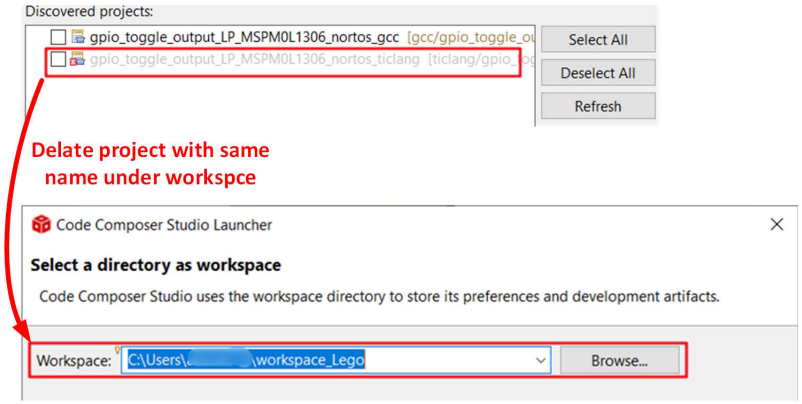


図 2-29. 重複したプロジェクトの削除

3. インポート後、左側にプロジェクトが表示され、自動的に README.md が開きます。TI では、最初に README.MD ファイルを読むことを推奨します。このファイルには、この例の目的ハードウェア構成が記載されています。

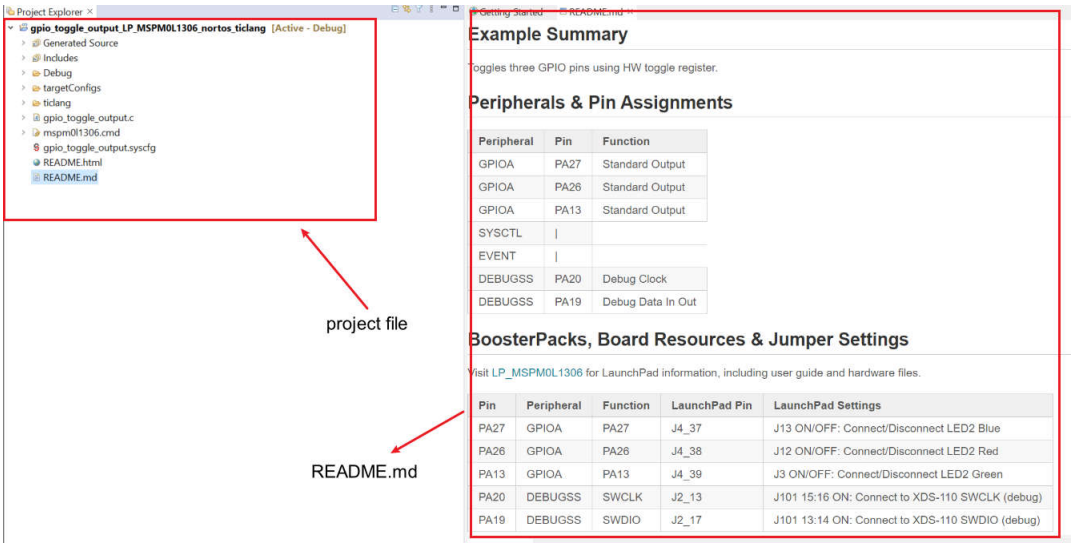


図 2-30. プロジェクトと README.md

4. 図 2-31 に、プロジェクトで重要なファイルを示します。

図 2-31. CCS プロジェクトの概要

5. STM の開発と同様に、TI ではチップ ビューもサポートしています。.syscfg ファイルをダブルクリックすると SysConfig が開き、必要な周辺機能をグラフィカルなインターフェースで設定できます。また、TI では、SysConfig の MCU ビューの使用を推奨しています。これを使用すると STM8CubeMAX の MCU、MPU パッケージと同様に、ソフトウェアエンジニアと連携して最初にピン機能を決定することができます。
6. コードや SysConfig のサンプルをもとに、ユーザーはプロジェクトを改良したり、Ti.com で公開されているデバイス固有の TRM（テクニカルリファレンスマニュアル）またはアプリケーションノートを参照して、プロジェクトを編集することができます。
7. サードパーティのライブラリを追加する場合、以下の手順に従ってください。まず、関連ファイルをプロジェクトに追加します (図 2-32 を参照)。

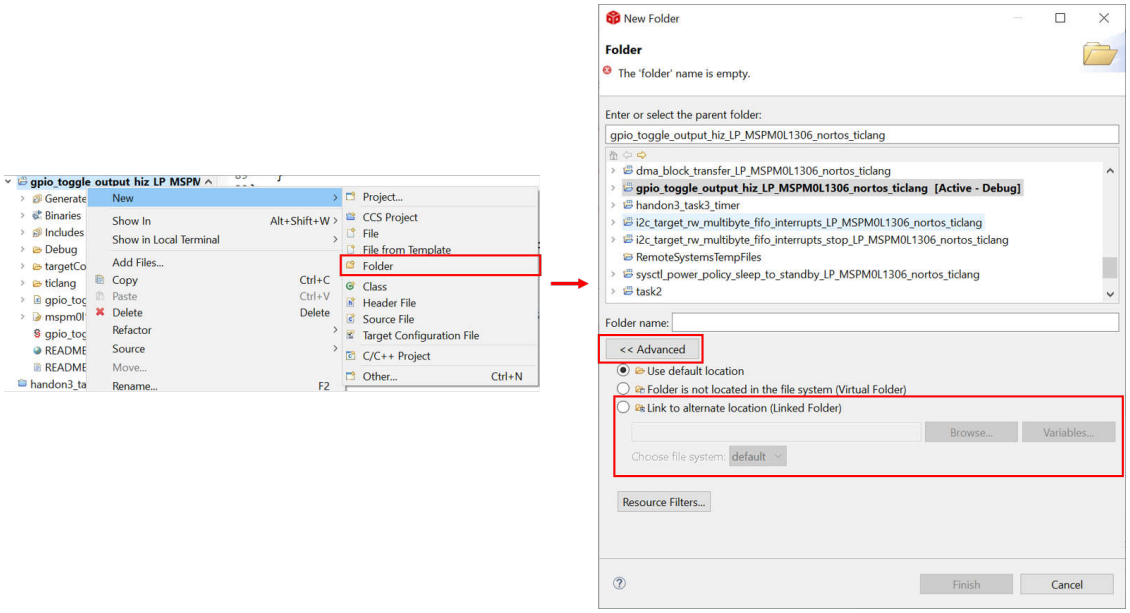


図 2-32. 関連ファイルの追加

次に、コンパイラにヘッダファイルを追加するよう指示するための設定が必要です。

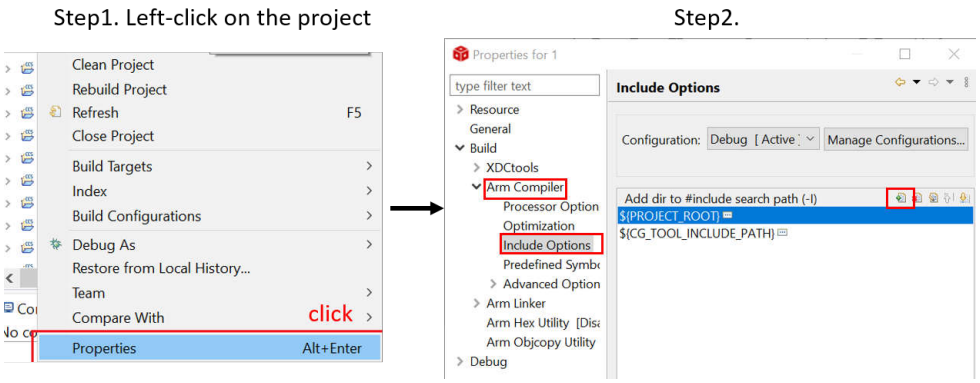


図 2-33. オプションセットを含める

8. ソフトウェアの評価を完了したら、メインツールバーの*ビルド*アイコンをクリックします (図 2-34 を参照)。*Build Finished* と表示されれば、コンパイルが正常に完了したことを意味します。

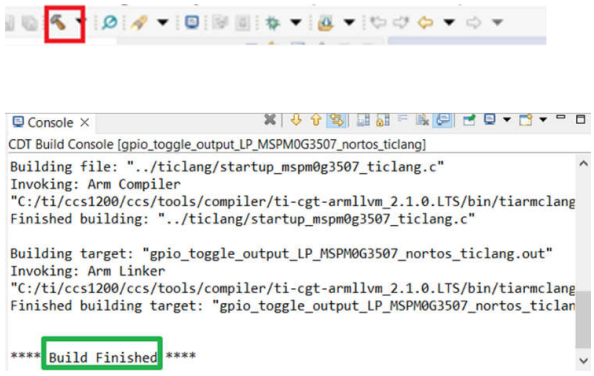


図 2-34. ビルド成功

2.2.5 ステップ 5.PCB ボードの設計

1. 設計パッケージを入手するには、Ti.com にアクセスし、目的のデバイスページに移動します。MSPM0L1304 を例にします。
2. 「Design and development」(設計と開発) → CAD/CAE シンボルをクリックします。必要に応じて異なるパッケージモデルを選択してダウンロードします。

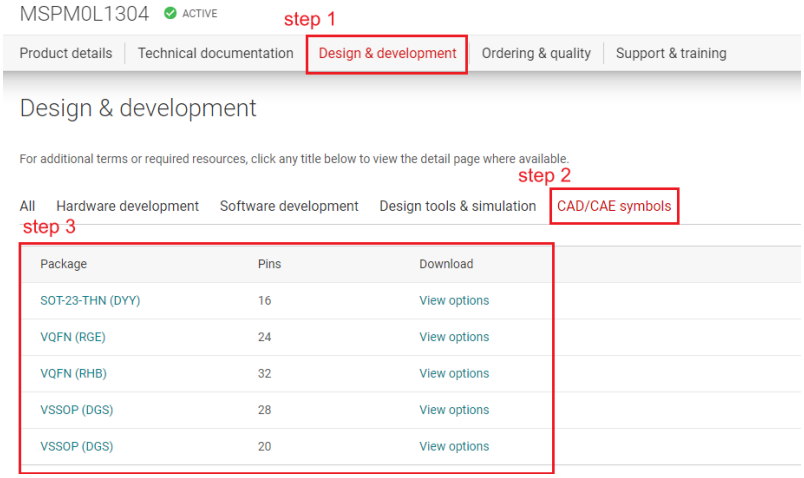


図 2-35. Ultra Librarian ツールへの入口

3. 自作の基板に組み込むための MSPM0 ハードウェア設計。図 2-36 に、最小システム設計例を示します。

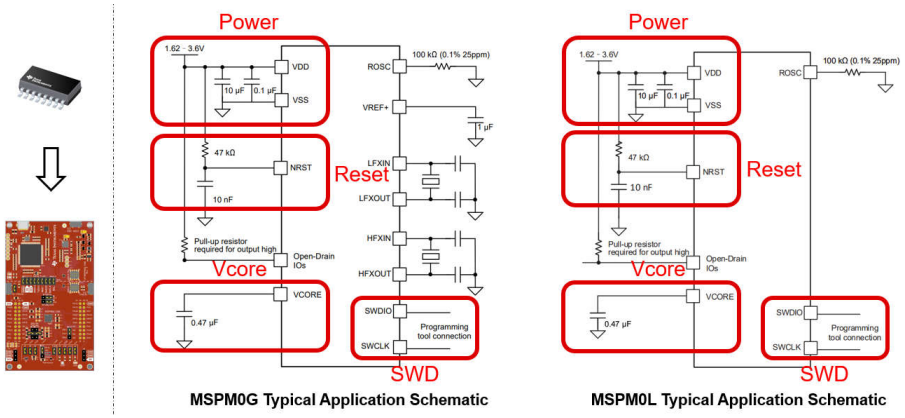


図 2-36. MSPM0 最小システム

最小システムにする場合、次の点に注意する必要があります:

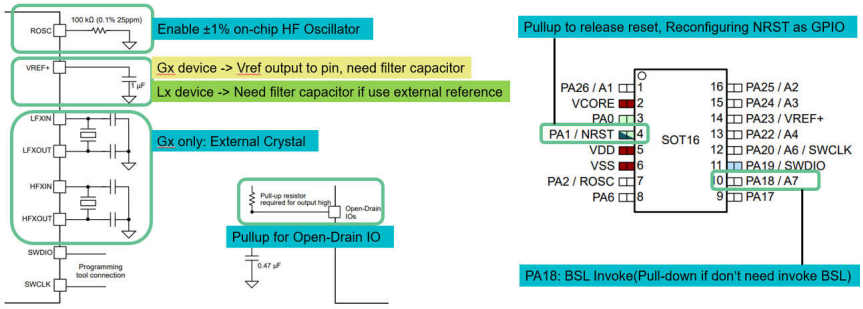


図 2-37. MSPM0 最小システムの注意事項

ハードウェア開発の詳細については、次の資料を参照してください:

- 『MSPM0 G シリーズ MCU ハードウェア開発ガイド』
- 『MSPM0 L シリーズ MCU ハードウェア開発ガイド』

2.2.6 ステップ 6.量産

1. CCS を使用して本番運用ファイル (.bin/.txt/…) を生成します。

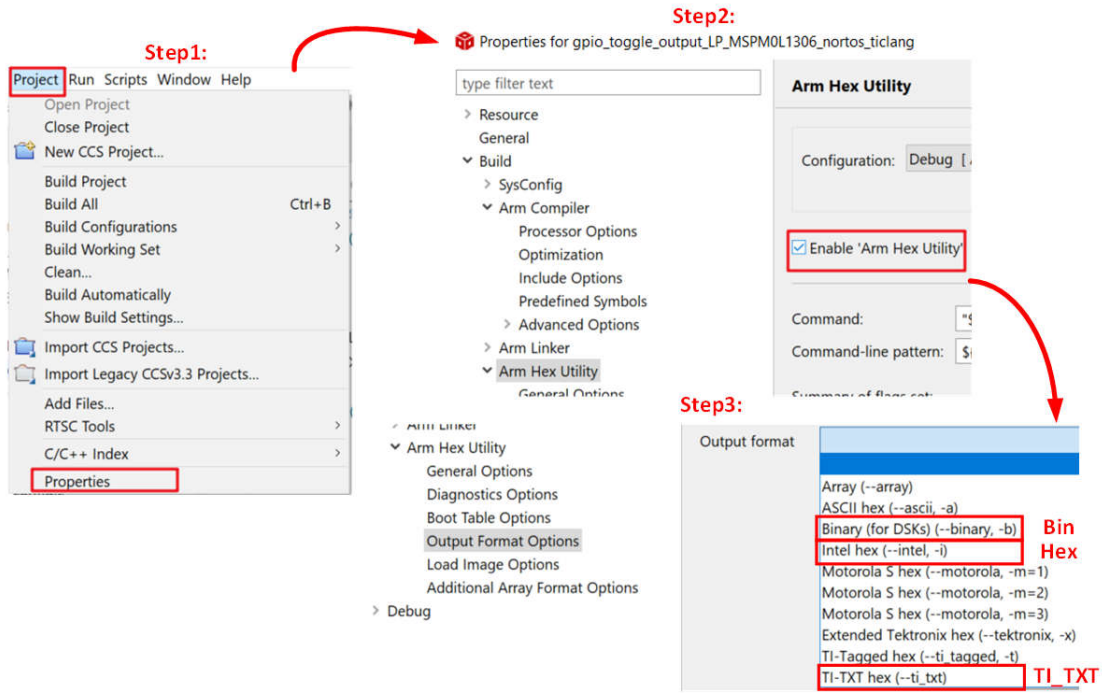


図 2-38. プログラム ファイルの作成

2. MSP デバイスをプログラムするプログラマ/デバッガを選択します。

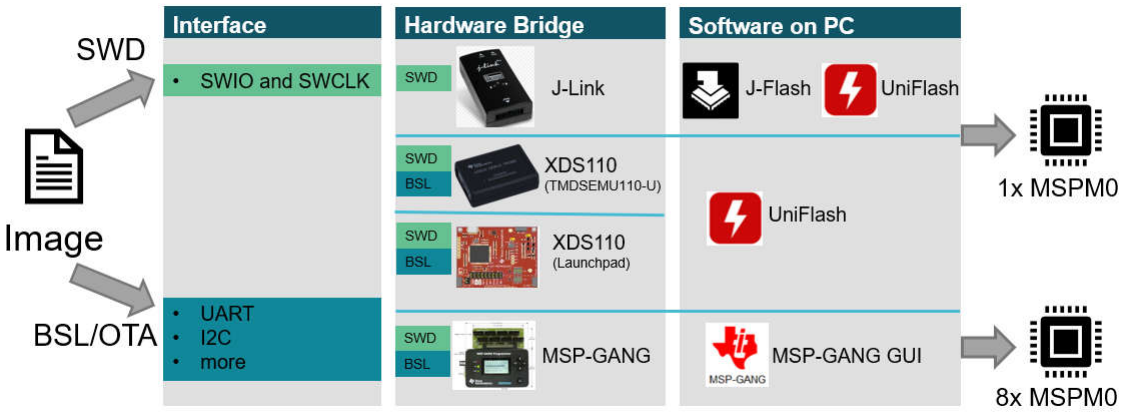


図 2-39. プログラムソフトウェアおよびツール

MSP-GANG および J-LINK の使用方法については、以下を参照してください:MSPM0 デザイン フロー ガイド。

デバッグの詳細については、以下を参照してください:デバッグおよびプログラミング ツール ガイド。

2.3 例

MSPM0 の設計フローを以下に示します。この例では、PWM を使用して LED を駆動することを目的としています。

1. 適切な MSPM0 MCU を選択し、ハードウェアを選択して EVM を注文します。ここでは、Launchpad MSPM0L1306 を使用します。
2. CCS と SDK を設定します。詳細については、セクション 2.2 を参照してください。
3. コードのインポート。

   環境の準備ができた時点で、CCS にコードをインポートできます。この例では、タイマを使用して PWM を制御します。最初に STM8 と MSPM0 のタイマーモジュールの違いを理解し、MSPM0 の SDK 内で類似する例を選びます。

   SDK 内で最も近い例は、おそらく *timx_timer_mode_pwm_center_stop* です。類似の例が見つかったら、CCS を開き、「Project > Import CCS Project...」の順に選択してサンプル コードをインポートし、MSPM0 SDK のサンプル フォルダに移動します。

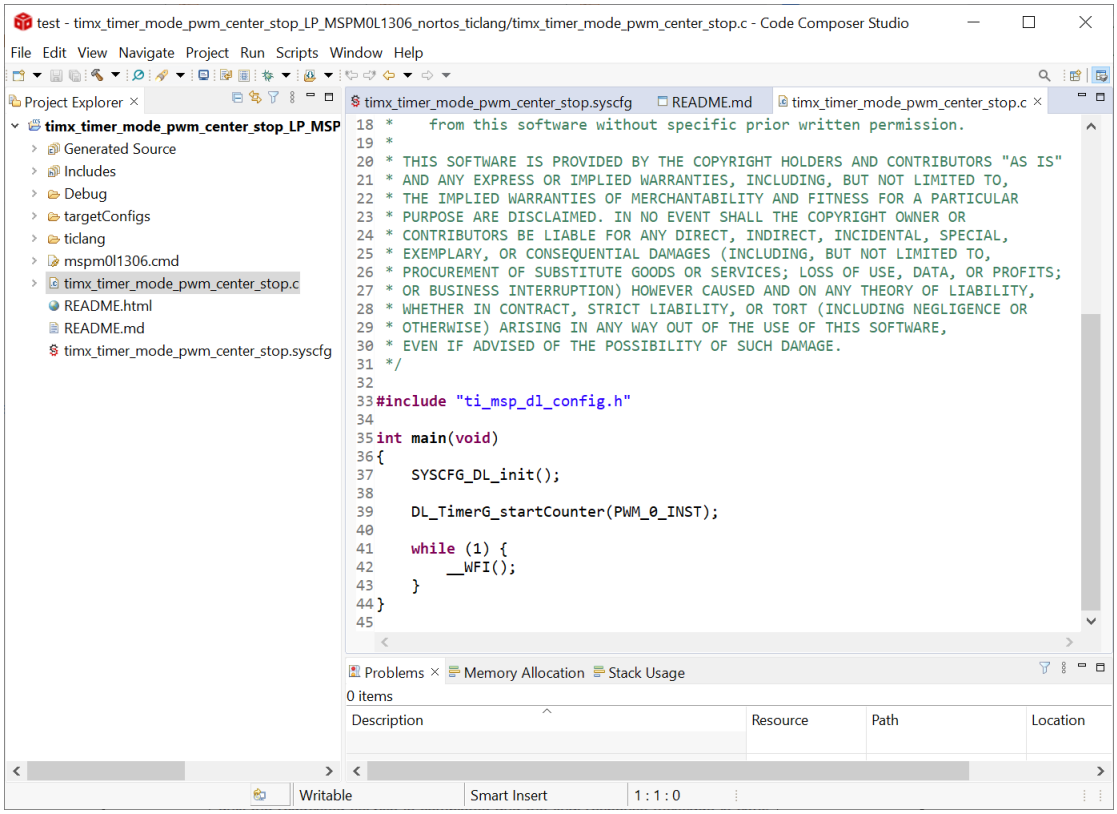


図 2-40. コード サンプル ファイル

4. プロジェクトを編集します。

   SysConfig の設定内容を確認するには、.syscfg ファイルを開きます。PWM を生成するために TIMER-PWM セクションを選択します (図 2-41 を参照)。PWM のクロック設定 (周波数やデューティサイクル) を確認します。この例では、PWM の周波数が 2.7Hz、デューティサイクルが 75% です。目的のデューティサイクル (50% など) を入力することでデューティサイクルを簡単に変更でき、カウンタ比較値が自動的に変更されます。

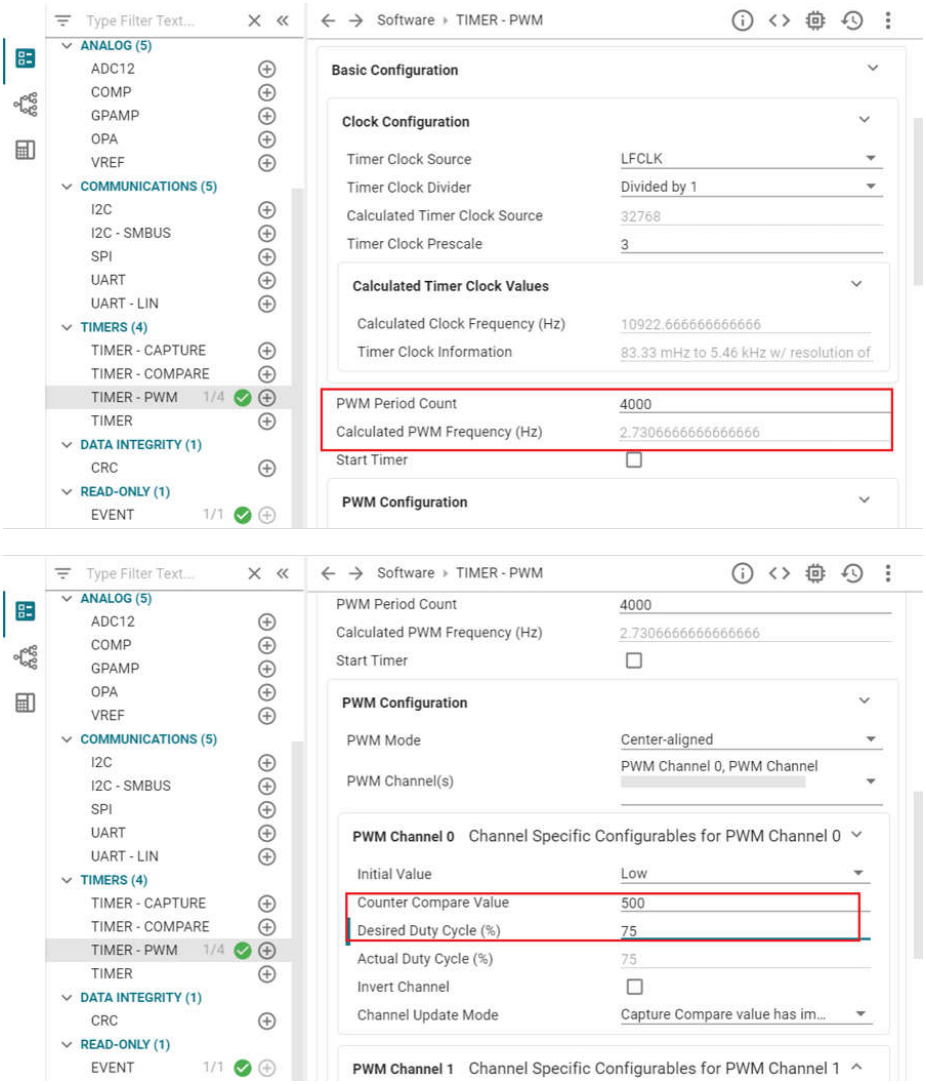


図 2-41. SysConfig 内の PWM 構成

各機能モジュールの詳細については、各項目の横にある「?」をクリックしてください。

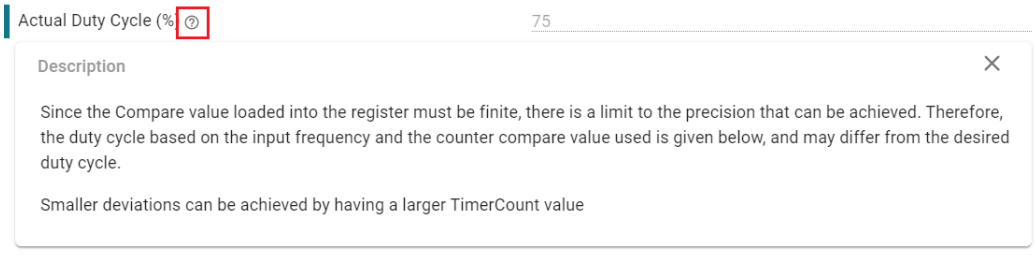


図 2-42. 各項目の詳細情報の取得

また、右上にあるチップ アイコンをクリックし、PWM で強調表示されているピンをチェックして、TIMER-POWER モジュールの他の機能と使用中のピンを確認します。

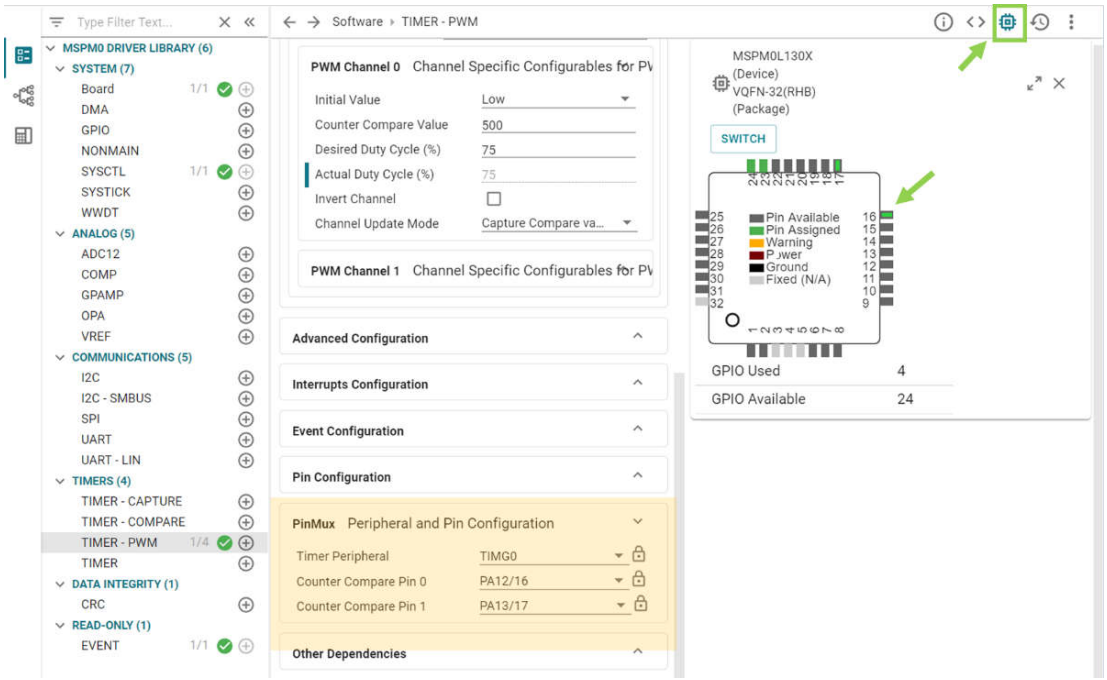


図 2-43. ピン構成

プロジェクトを保存して再ビルドすると、SysConfig によって、図 2-44 内のファイルが更新されます。この時点で、サンプル ハードウェア構成が変更され、移植対象の元のソフトウェアのすべての機能が一致するようになりました。

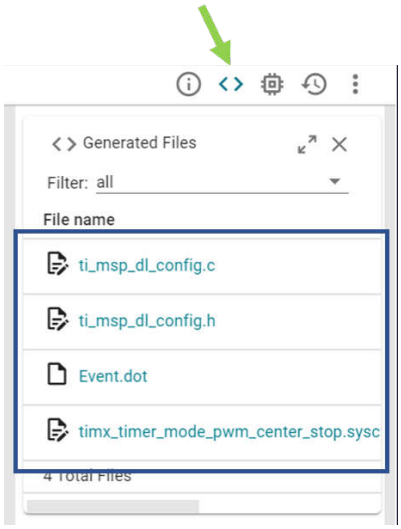


図 2-44. SysConfig ファイルの更新

あとは、アプリケーションレベルのソフトウェアを確認するだけです。この例では、SDK コードと同様に PWM 波形を生成するため、.c ファイルを変更する必要はありません。

5. ハードウェア設定。

   LaunchPad をパソコンに接続します。ピン構成に従って、DuPont ケーブルを使用して PA12 を LED ピンに接続します。

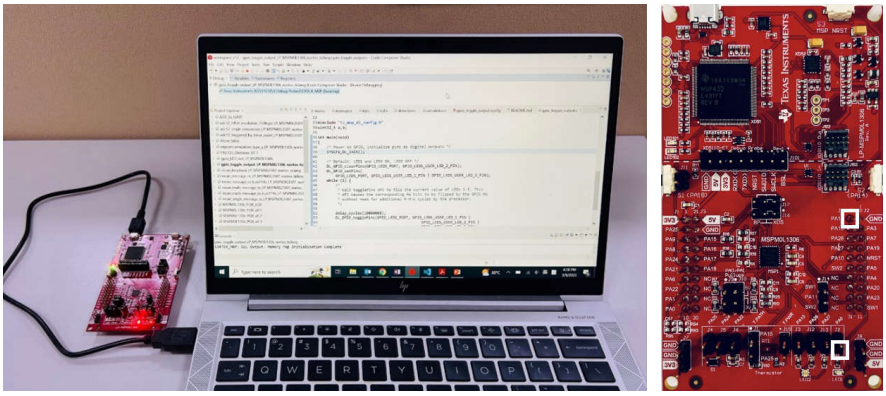


図 2-45. ハードウェア設定

6. デバッグと検証。

   デバッグアイコンをクリックして、デバッグを開始します。行番号の前のスペースをダブルクリックするか、1 行のコード *__BKPT()*を追加することで、ブレークポイントを設定できます。

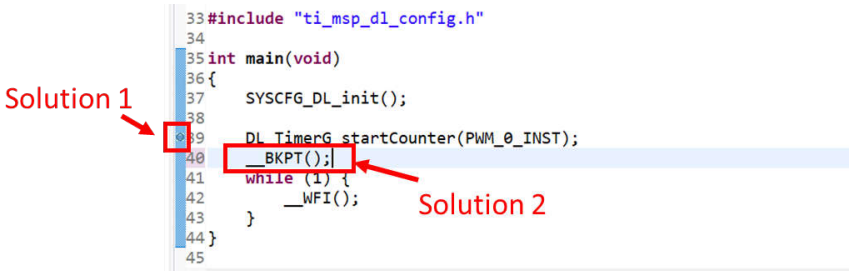


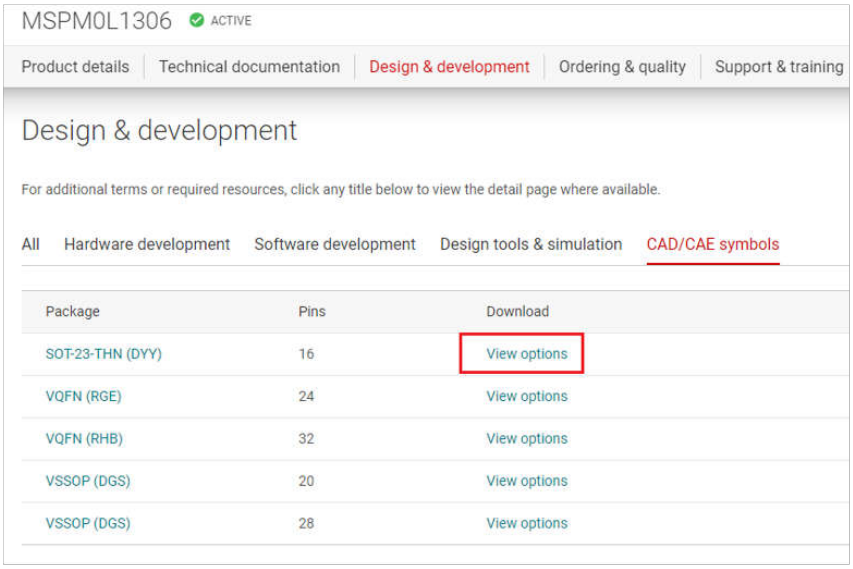
図 2-46. ブレークポイントソリューションの追加

   デバッグ機能 (詳細はセクション 2.2.2.2 を参照) を使用して、手順が正しく実行できるかどうかを検証します。デバッグ中にコードをステップ実行すると LED の点滅を確認できます。

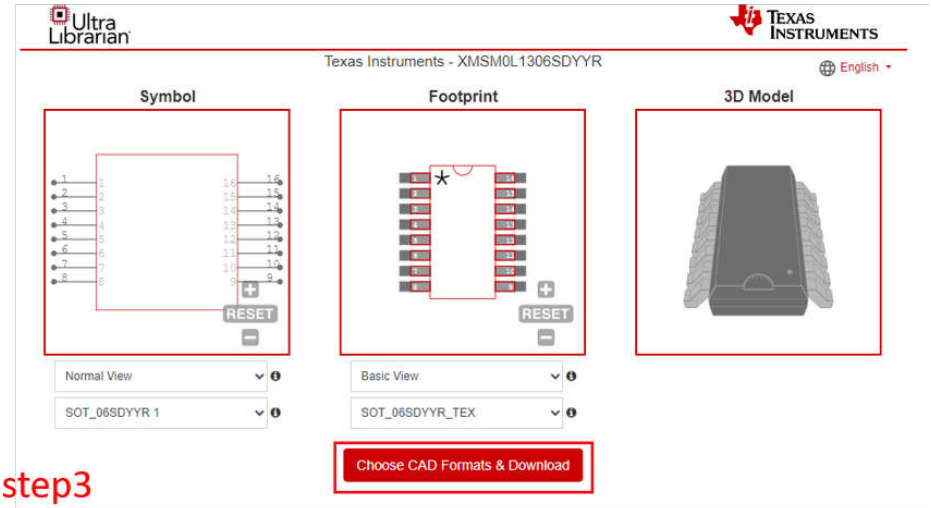
7. PCB ライブラリを生成し、Altium Design にインポートします。

   具体的な手順を図 2-47 に示します。MSPM0 デバイスページの下にある Ultra Librarian ツールへの入り口に移動します (詳細はセクション 2.2.5 を参照)。*表示オプション*をクリックします。希望する CAD 形式とピン配列を選択し、Altium 設計ライブラリファイルを入手します。

step1



step2



step3

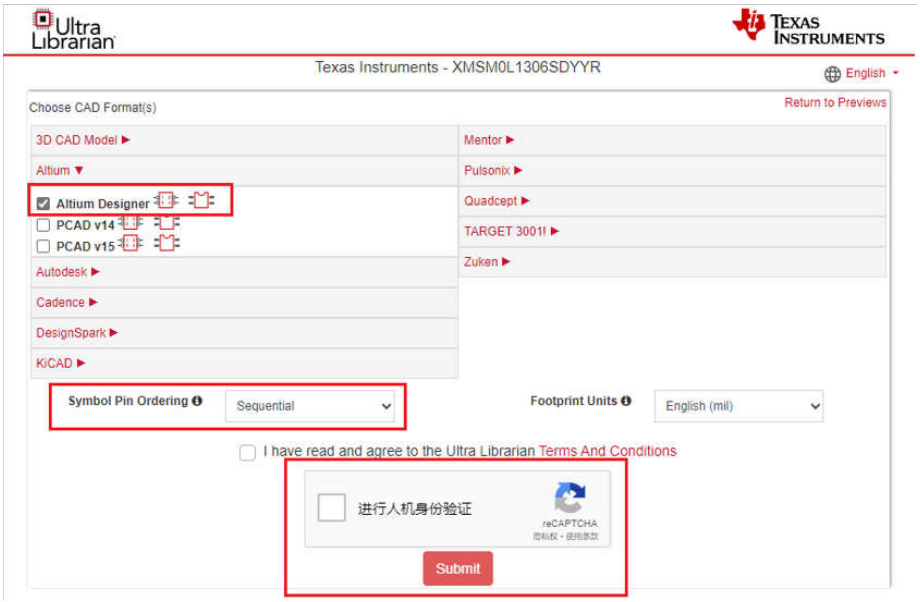


図 2-47. Ultra Librarian ツールのダウンロード

ライブラリをダウンロードしたら、Altium Designer のスクリプトを実行し、PCB ライブラリと回路図ライブラリを生成します (図 2-48 を参照)。

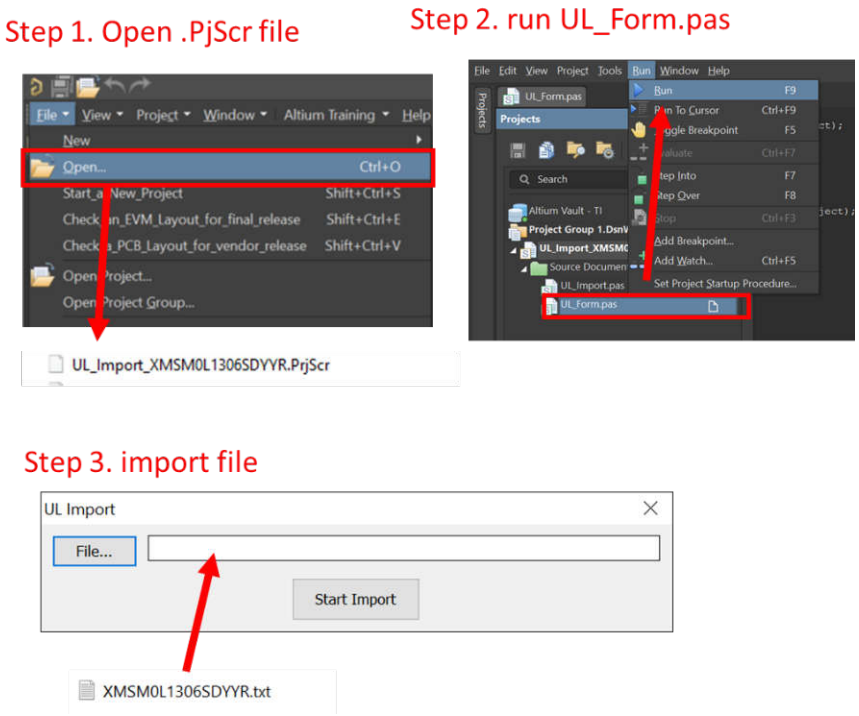


図 2-48. Altium Designer スクリプトを実行

手順を完了すると、同じソースフォルダ内に以下の新しいファイルが生成されます。

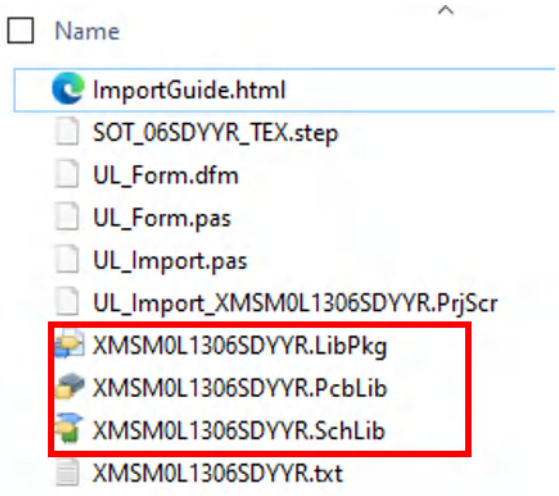


図 2-49. PCB ライブラリと回路図ファイル

最後に、AD ライブラリにこれらをインポートします (図 2-50 を参照)。これをもとに、回路図と PCB を設計できます。

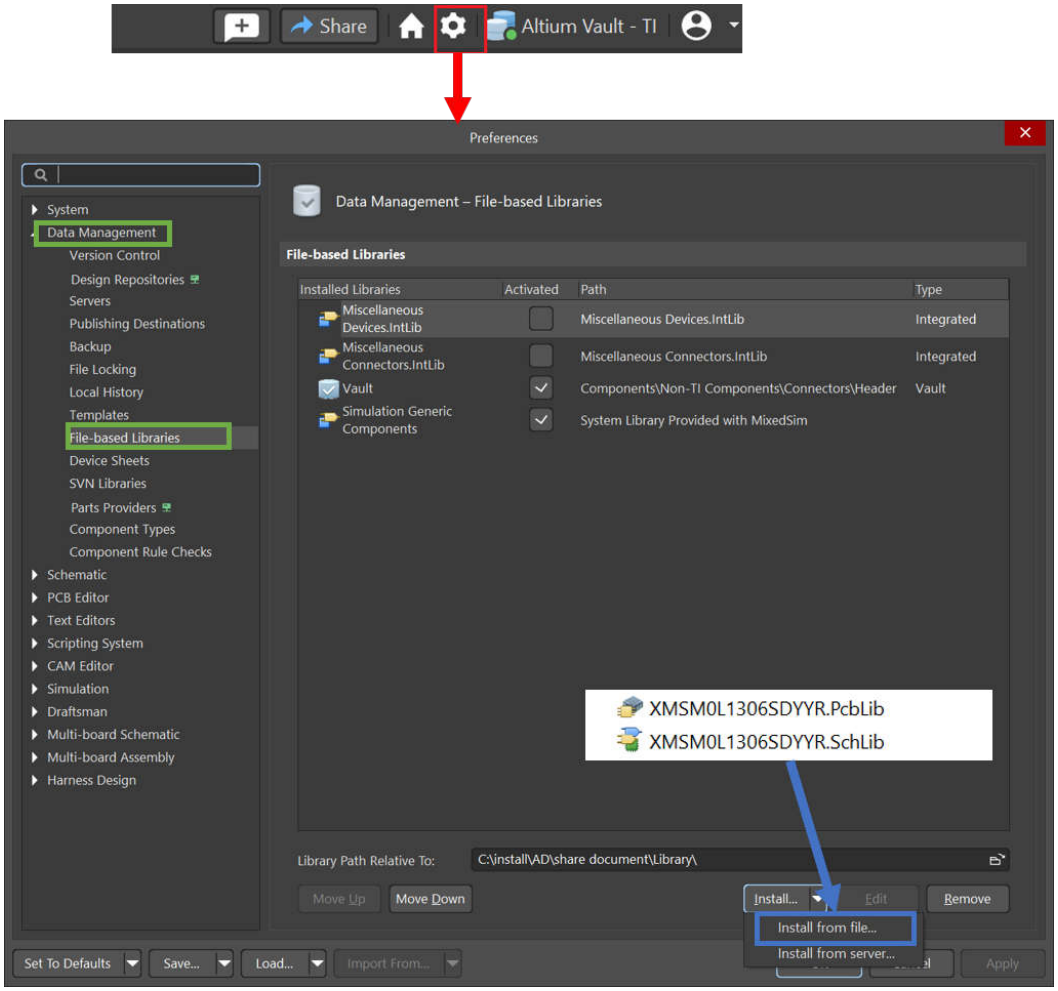


図 2-50. ライブラリのインポート

8. MSPM0 での設計。
9. 量産。

3 コア アーキテクチャの比較

3.1 CPU

MSPM0 ファミリは、ARM Cortex M0+ CPU コアアーキテクチャをベースにしています。STM8 ファミリは、STM8 CPU コアアーキテクチャをベースにしています。表 3-1 に、MSPM0 ファミリの CPU と STM8 の CPU の主な特長を比較して紹介しています。

表 3-1. CPU 機能セットの比較

特長	STM8L、STM8S	MSPM0C、MSPM0L および MSPM0H
アーキテクチャ	拡張 STM8 CPU コア	Arm Cortex M0+
データ バス幅	8 ビット	32 ビット
命令セット	複雑な命令セット	縮小命令セット
命令の数	80	56
乗算命令	MUL (8×8)	MULS (32×32)
除算命令	DIV (16÷8)、DIVW (16÷16)	MATHACL は、32 ビット除算をサポートしています(1)
パイプライン	3 ステージ	2 ステージ
動作周波数 (最大)	16MHz または 24MHz(2)	24MHz または 32MHz(3)
DMA	あり	あり
Coremark/MHz	使用不可(4)	2.39(5)

(1) MSPM0Gxx シリーズには 32 ビット除算速度を向上させる演算アクセラレータ (MATHACL) が搭載されています。
(2) STM8Lxx の最大動作周波数は 16MHz、STM8Sxx は 24MHz です。
(3) MSPM0Cxx の最大動作周波数は 24MHz、MSPM0Lxx は 32MHz です。
(4) STM8 の Coremark スコアは、st.com および eembc.com では公開されていません。
(5) Coremark スコアは、ARM 公式 Web サイトの *Arm Cortex-M0+ プロセッサのデータシート*から取得しています。

3.2 組み込みメモリの比較

3.2.1 フラッシュ メモリと EEPROM の特長

MSPM0 および STM8 ファミリの MCU には、実行可能なプログラム コードとアプリケーション データの保存に使用される不揮発性フラッシュ メモリと EEPROM が搭載されています。表 3-2 に、フラッシュ メモリと EEPROM の機能を示します。すべての機能がすべてのデバイスにあるわけではないことに注意してください。詳細については、デバイス固有のデータシートを参照してください。

表 3-2. フラッシュ メモリと EEPROM の特徴

特長	STM8L、STM8S	MSPM0L、MSPM0C および MSPM0H	
フラッシュ メモリ	STM8Lxx は 2 KB～64 KB、 STM8Sxx は 4 KB～128 KB	MSPM0Lxx は 8 KB～64 KB、 MSPM0Cxx は 8 Kb または 64 KB、MSPM0Hxx は 32KB～64KB	
EEPROM	最大 2 KB	フラッシュ メモリを使用した EEPROM エミュレーション	
メモリ構成	ブロックサイズ (64 B/128 B) ページサイズ (ブロックのセット)	ワードライン (128B) セクタサイズ (1 KB) バンクサイズ (可変):デバイスは最大 256KB (1 バンク)	
ウェイト状態	0 (f_{CPU} ＜ 16MHz) 1 (f_{CPU} ＞ 16MHz)	0 (MCLK, CPUCLK ＜ 24MHz) 1 (MCLK, CPUCLK ＜ 48MHz)	
シングル ワード サイズ	32 ビット	64 ビット (ECC 付きで 72 ビット)	
プログラミング モード	バイト、シングル フラッシュ ワード、ブロック	分解能	シングル フラッシュ ワード、32、16、または 8 ビット
		マルチワード	2、4、8 ワード (最大 64 バイト)
消去	ブロック消去	セクタ消去 バンク消去 (最大 256KB)	

表 3-2. フラッシュ メモリと EEPROM の特徴 (続き)

特長	STM8L、STM8S		MSPM0L、MSPM0C および MSPM0H
エラー コード訂正	対応		対応
書き込み保護	あり		あり、静的と動的
読み取り保護	あり		あり
消去および書き込みサイクル	プログラム メモリ	100	100K (下位 32KB) または 10k (32KB 超過)
	データ メモリ	100k	

前の表に示したフラッシュ メモリ機能に加えて、MSPM0 フラッシュ メモリには以下の機能もあります:

- 電源電圧範囲全体にわたって、インサーキット プログラムと消去がサポートされています。
- プログラム間の電圧生成。

3.2.2 フラッシュ メモリと EEPROM の構成

フラッシュ メモリ と EEPROM は、1 つ以上の論理メモリ領域にマップされ、アプリケーションで使用できるようにシステム アドレス空間が割り当てられます。

3.2.2.1 フラッシュ メモリと EEPROM のリージョン

表 3-3 に、STM8 デバイスおよび MSPM0 デバイスのフラッシュ メモリと EEPROM のリージョンを示します。

表 3-3. フラッシュ メモリと EEPROM のリージョン

STM8L および STM8S		MSPM0L、MSPM0C および MSPM0H	
メイン プログラム エリア	アプリケーション コード。	**MAIN**	アプリケーション コードおよびデータ。
動作バイト	デバイス ハードウェア機能の構成とメモリ保護。	**NONMAIN**	BCR の構成。
			BSL の構成。
独自のコード領域 (PCODE)(1)	ペリフェラルの制御に使用される独自ソフトウェア ライブラリの保護。	**FACTORY**	デバイス ID およびその他のパラメータ。
ユーザー ブート エリア (UBC)(2)	リセット ベクタと割り込みベクタ。		
データ EEPROM	アプリケーション データ。	**DATA**(3)	データまたは EEPROM エミュレーション。

(1) PCODE は、低、中+、および高密度デバイスでのみ使用できます。
(2) STM8Sx03xx、STM8S001xx、STM8L101xx、STM8L001xx には、ブートローダーが内蔵されていません (マイクロコントローラに ROM ブートローダーは実装されていません)。これらのデバイスを使用する場合、ユーザー自身がブートローダー コードを作成し、そのコードを UBC プログラム領域に保存する必要があります。
(3) 1 つのバンクを持つ MSPM0 デバイスは、FACTORY、NONMAIN、および MAIN 領域を BANK0 (存在する唯一のバンク) に実装しており、データ領域は利用できません。複数のバンクを持つ MSPM0 デバイスは、FACTORY、NONMAIN、および MAIN 領域も BANK0 に実装していますが、MAIN 領域または DATA 領域を実装できる追加のバンク (BANK1 から BANK4) が含まれています。

3.2.2.2 MSPM0 の NONMAIN メモリ

NONMAIN フラッシュ メモリには、FLASHSWP0 や FLASHSWP1 (静的書き込み保護ポリシー) など、デバイスをブートするために BCR および BSL によって使用されるコンフィギュレーション レジスタが含まれています。この領域は、他の目的では使用されません。BCR と BSL にはどちらも構成ポリシーがあり、NONMAIN フラッシュ領域にプログラムされた値を変更することで、デフォルト値 (開発および評価時の標準値) をそのままにするか、特定の目的 (量産プログラミング時の標準値) に変更することができます。

3.2.3 内蔵 SRAM

MSPM0 および STM8 ファミリの MCU には、アプリケーション データの保存に使用される SRAM が搭載されています。表 3-4 に、SRAM の機能の比較を示します。詳細は、デバイス固有のデータシートを参照してください。

表 3-4. SRAM 機能の比較

特長	STM8L、STM8S	MSPM0L、MSPM0C および MSPM0H
SRAM メモリ	STM8L バリュー ライン：1 KB～4 KB STM8L101：1.5 KB STM8Sxx：1KB～6KB	MSPM0Lxx：2 KB～4 KB MSPM0Cxx：1 KB またはまたは 8KB MSPM0Hxx：8KB
パリティ チェック	非対応	MSPM0Lxx：対応(1)、MSPM0Cxx：非対応、MSPM0Hxx：非対応
ECC	非対応	MSPM0Lxx：対応(2)、MSPM0Cxx：非対応、MSPM0Hxx：非対応
書き込み保護 (RAM ガード)	非対応	対応

(1) MSPM0Lx22x と MSPM0L111x のみサポートしています。
(2) MSPM0Lx22x のみサポートしています。

MSPM0 MCU には、低消費電力の高性能 SRAM が搭載されており、デバイスでサポートされている CPU 周波数範囲全体にわたってゼロ ウェイト ステートに対応します。SRAM は、コードに加えて、呼び出しスタック、ヒープ、グローバル データなどの情報を格納するために使用できます。SRAM の内容は、実行、スリープ、停止、スタンバイ動作モードでは完全に保持されますが、シャットダウン モードでは失われます。書き込み保護メカニズムが搭載されているため、アプリケーションは 1KB の分解能で下位 32KB の SRAM を動的に書き込み保護できます。32KB 未満の SRAM を搭載したデバイスでは、SRAM 全体に対して書き込み保護が提供されます。書き込み保護は、CPU または DMA によってコードが意図せず上書きされることに対してある程度の保護を提供するため、実行可能コードを SRAM に配置する場合に役立ちます。SRAM にコードを配置すると、ゼロ ウェイト状態動作と低消費電力を実現することで、重要なループの性能を向上できます。

3.3 電源投入とリセットの概要と比較

STM8 デバイスと MSPM0 デバイスは共に最小動作電圧があり、デバイスまたはデバイスの一部をリセット状態に保持することでデバイスを正しく起動させるためのモジュールが搭載されています。表 3-5 に、2 つのファミリ間でどのように行われ、どのモジュールが、ファミリ間で電源投入プロセスとリセットを制御するかを比較したものを示します。

表 3-5. 電源投入の概要と比較

STM8		MSPM0	
パワーオン リセット (POR)	立ち上がり検出：V_{DD}>V_{POR}、POR 状態が解除され、BOR が動作を開始します。	**パワーオン リセット (POR)**	立ち上がり検出：V_{DD} が POR+ を上回ると、POR 状態が解除され、バンドギャップ参照と BOR が開始されます 立ち下がり検出：V_{DD} が POR- を下回ると、デバイスは POR 状態に固定されます
パワーダウン リセット (PDR)	立ち下がり検出：V_{DD} が V_{PDR} を下回ると、PDR がデバイスをリセット状態に保持します。		

表 3-5. 電源投入の概要と比較 (続き)

STM8		MSPM0	
ブラウンアウト リセット (BOR)(1)	立ち上がり検出:V_{DD} が V_{BOR+} を上回ると、BOR 状態が解除され、デバイスはブートプロセスを継続します。 立ち下がり検出:V_{DD} が V_{BOR-} を下回る、BOR 状態が生成されます。 BOR には選択可能な 5 つのレベルがあります。	**ブラウンアウト リセット (BOR)- 0 レベル**(2)	立ち上がり検出:V_{DD} が BOR0+ を上回ると、 デバイスはブートプロセスを継続し、PMU が起動します 立ち下がり検出:V_{DD} が BOR0- を下回ると、 デバイスは BOR 状態に固定されます。
		ブラウンアウト リセット (BOR)- 1～3 レベル(2)	立ち下がり検出: 1) V_{DD} が BORx- (x=1、2、3) を下回ると、割り込み要求が発生し、BOR 回路は自動的にしきい値レベルを BOR0 に切り替えます。 2) V_{DD} が BOR0- を下回ると、デバイスは BOR 状態に固定されます
プログラマブル電圧検出器 (PVD)(3)	立ち上がり検出:V_{DD} が V_{PVD} を上回ると、PVD イベントが生成されます。 立ち下がり検出:V_{DD} が V_{PVD} を下回ると、PVD イベントが生成されます。PVD には選択可能な 7 つのレベルがあります。	**該当なし**	該当なし
RTC のリセット	RTC および関連レジスタは、システム リセットまたはパワーオン リセットによりリセットされます。	**RTC のリセット**	RTC および関連クロックは、BOOTRST、BOR、または POR によってリセットされます

(1) スタートアップ時には、最低動作電圧を確保するために BOR をレベル 0 に設定する必要があります。BOR は電源投入時に常にアクティブであり、アプリケーションが動作可能な電圧に達するまで MCU をリセット状態に保ちます。パワーダウン時に BOR がディセーブルになっている場合、リセット閾値は V_{PDR} になり、VDD の最小値が確保されます。

(2) 四つの選択可能な BOR スレッショルド レベル (BOR0 ～ BOR3) があります。起動中、BOR 閾値は常に BOR0 (最小値) になり、デバイスは常に指定された V_{DD} の最小値で起動します。ブート後、ソフトウェアによって BOR 回路の閾値レベルを別の (より高い) レベルに再設定することも可能です。

(3) STM8L001xx、STM8L101xx、STM8S シリーズのマイクロコントローラには PVD が搭載されていません。

図 3-1 に、MSPM0 のリセット機能を示します。MSPM0 デバイスには 5 つのリセット レベルがあります:パワーオン リセット (POR)、ブラウンアウト リセット (BOR)、ブート リセット (BOOTRST)、システム リセット (SYSRST)、CPU リセット (CPURST)。

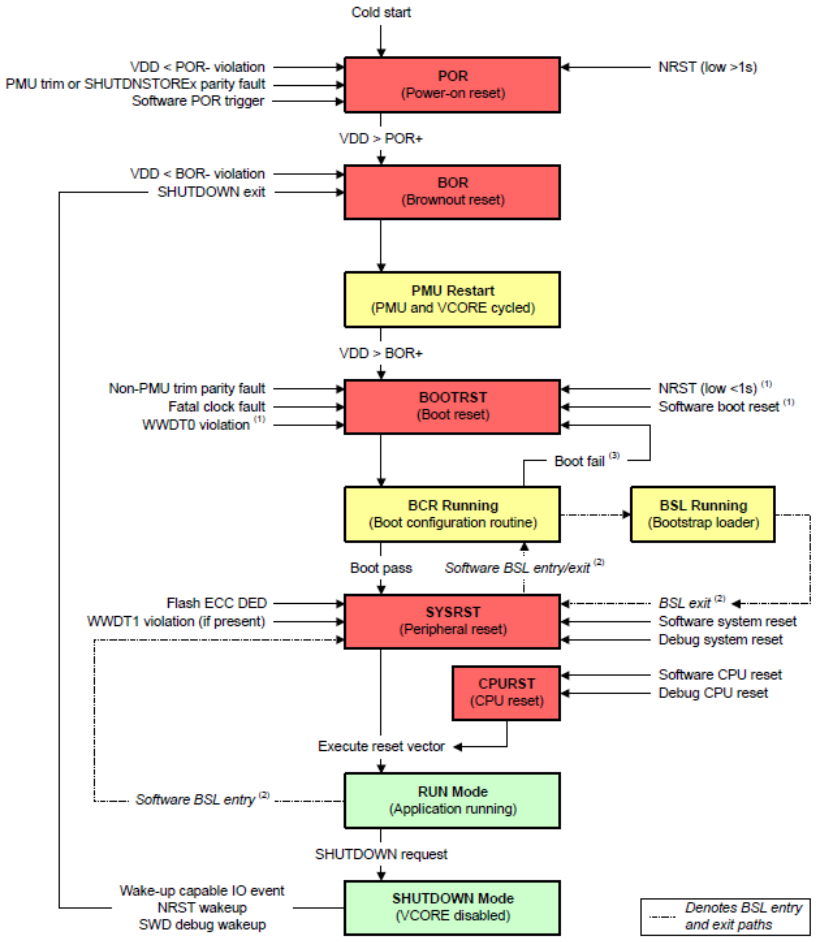


図 3-1. MSP リセット機能

3.4 クロックの概要と比較

3.4.1 発振器

STM8 および MSPM0 デバイスには、低システムコストと低消費電力のために、内部および外部の多くのタイプのクロックソースが用意されています。表 3-6 に、STM8 および MSPM0 デバイスの各種クロック ソースを示します。すべてのデバイスにすべてタイプのクロックソースがあるわけではないことに注意してください。詳細については、デバイス固有のデータシートを参照してください。

表 3-6. 発振器の比較

タイプ		STM8L	STM8S	MSPM0L	MSPM0C	MSPM0H
内部発振器	高速	HSI：内部 16MHz RC 発振器	HSI：内部 16MHz RC 発振器	SYSOSC：内部発振器 (4MHz～32MHz)	SYSOSC：内部 24/32 MHz 発振器	SYSOSC：内部 32MHz 発振器
	低速	LSI：内部 38kHz RC 発振器	LSI：内部 128kHz RC 発振器	LFOSC：内部 32kHz 発振器	LFOSC：内部 32kHz 発振器	LFOSC：内部 32kHz 発振器
外部発振器	高速	HSE：1～16 MHz 外部水晶振動子	HSE：1～24 MHz 外部水晶振動子	4MHz～32MHz(1)	4MHz～32MHz(1)	該当なし
	低速	LSE：32.768kHz 外部水晶振動子	該当なし	32kHz(1)	32kHz(1)	該当なし

(1) MSPM0Lx22x、MSPM0C1105、MSPM0C11056 のみがサポートされています。

3.4.2 クロック信号の比較

異なるクロック信号は分周して他のクロックを供給し、多数のペリフェラルに分配することができます。

表 3-7. クロック信号の比較

クロックの説明		STM8L クロック	STM8S クロック	MSPM0L、MSPM0C、MSPM0H クロック
外部デジタル クロック入力	高周波	外部ソース：最大 16MHz(1)	HSE 外部：最大 24MHz(1)	Lx22x および C1106：32MHz をサポート
	低周波	外部ソース：32.768kHz(1)	該当なし	Lx22x および C1106：32KHz をサポート
メイン クロックの高周波ソース		HSI、HSE	f_{HSE}、f_{HSIDIV}	SYSOSC
メイン クロックの低周波数ソース		LSI、LSE	f_{LSI}	LFCLK (32kHz 固定)
メイン システム クロック		SYSCLK、f_{MASTER}	f_{MASTER}	MCLK、ULPCLK (BUSCLK) (2)
ソース CPU		SYSCLK、f_{MASTER}	f_{CPU}	CPUCLK
ほとんどのペリフェラルハードウェア用のクロック		PCLK (SYSCLK)、f_{MASTER}	f_{MASTER}	MCLK、ULPCLK(2)
ペリフェラル固有のクロック		BEEPCLK、IWDGCLK、RTCCLK、f_{LSI}、$f_{HSI/2}$ (3)	該当なし	ADCCLK
固定周波数クロック		該当なし	該当なし	MFCLK：4Mhz (MCLK、ULPCLK に同期)

(1) 外部クロックソースを使用する場合、HSE 水晶振動子と LSE 水晶振動子をオフにする必要があります。STM8L001xx および STM8L101xx ファミリは、外部デジタル クロック入力をサポートしていません。

(2) MCLK は PD1 のメイン システム クロックであり、ULPCLK は MCLK から派生した PD0 のメイン システム クロックです。PD1 (電源ドメイン 1) には、CPU サブシステム、メモリ インターフェイス、高速ペリフェラルが含まれています。PD0 (電源ドメイン 0) には、低速、低消費電力ペリフェラルが含まれています。

(3) STM8L001xx および STM8L101xx ファミリの f_{LSI} は、AWU、BEEP、SWIM、IWDG のソースとしてのみ使用されます。$f_{HSI/2}$ は、SWIM のソースとしてのみ使用されます。

表 3-8. ペリフェラル クロック ソース

ペリフェラル	STM8L、STM8S	MSPM0L、MSPM0C、MSPM0H
UART/USART	SYSCLK、f_{MASTER}	SYSCLK、MFCLK、LFCLK
SPI	SYSCLK、f_{MASTER}	SYSCLK、MFCLK、LFCLK
I2C	SYSCLK、f_{MASTER}	BUSCLK、MFCLK
ADC	PCLK または PCLK/2[1]、f_{ADC} (f_{MASTER} を 2～18 で割った数)	ADCCLK (ULPCLK または SYSOSC から供給)
タイマ	SYSCLK、f_{MASTER}、$f_{MASTER/DIV}$	BUSCLK、MFCLK、LFCLK
コンパレータ	PCLK、f_{MASTER} [2]	BUSCLK
ウォッチドッグ	LSI、SYSCLK、f_{CPU} [3]	LFCLK

(1) STM8L001xx および STM8L101xx マイクロコントローラ ファミリには ADC は搭載されていません。
(2) STM8S シリーズのマイクロコントローラにはコンパレータはありません。
(3) LSI は、独立ウォッチドッグ (IWDG) のソースとして使用されます。SYSCLK または f_{CPU} は、ウィンドウ ウォッチドッグ (WWDG) のソースとして使用されます。

3.5 MSPM0 の動作モードの概要と比較

MSPM0 MCU には 5 つのメイン動作モード (電力モード) があり、アプリケーションの要件に基づいてデバイスの消費電力を最適化できます。消費電力を低減するためのモードは次のとおりです。RUN、SLEEP、STOP、STANDY、SHUTDOWN。CPU は RUN モードではコードをアクティブに実行しています。ペリフェラル割り込みイベントにより、デバイスを SLEEP、STOP、または STANDBY モードから RUN モードにウェークアップできます。SHUTDOWN モードでは、内部コア レギュレータが完全にディセーブルされ、消費電力が最小化されます。また、NRST、SWD、または特定の IO でのロジック レベルの一致によってのみウェークアップが可能です。RUN、SLEEP、STOP、STANDBY の各モードには、複数の構成可能なポリシー オプション (例:RUN.x) も含まれており、性能と消費電力のバランスを確保できます。

性能と消費電力のバランスをさらに高めるために、MSPM0 デバイスには次の 2 つの電力ドメインが実装されています。PD1 (CPU、メモリ、高性能ペリフェラル用) と PD0 (低速、低消費電力ペリフェラル用)。PD1 は、RUN モードと SLEEP モードで常に電源が供給されますが、他のすべてのモードではディセーブルになります。PD0 は、RUN、SLEEP、STOP、STANDBY の各モードで常に電源が供給されます。SHUTDOWN モードでは、PD1 と PD0 の両方がディセーブルになります。

3.5.1 動作モードの比較

表 3-9 に、STM8 デバイスと MSPM0 デバイスの簡単な比較を示します。

表 3-9. STM8 デバイスと MSPM0 デバイスの動作モードの比較

STM8		MSPM0		
動作モード	**説明**	**動作モード**		**説明**
実行モード	CPU およびペリフェラルは、システムまたは電源リセット後も通常どおり動作します。	**RUN**	0	MCLK と CPUCLK は、高速クロック ソース (SYSOSC) で動作します
低消費電力実行モード	CPU とペリフェラルは低速発振器 (LSI または LSE) で動作します。すべての割り込みをマスクする必要があります。		1	MCLK と CPUCLK は LFCLK から実行します (32kHz 時)。
			2	
ウェイト モード	CPU 動作が停止します。発振器はイネーブル状態を維持します。選択したペリフェラルが動作を継続します。WFI または WFE 命令を実行することで、実行モードからウェイト モードに移行します。	**SLEEP**	0	CPU 動作が停止します。SYSOSC はイネーブルのままです。LFOSC はイネーブルのままです。MCLK は高速クロック ソース (SYSOSC) で動作します。
			1	CPU 動作が停止します。SYSOSC はイネーブルのままです。LFOSC はイネーブルのままです。MCLK は LFCLK から実行されます。
低消費電力ウェイト モード	CPU 動作が停止します。低速発振器はイネーブル状態を維持します。選択したペリフェラルが動作を継続します。低消費電力実行モードでイベントの待機を実行すると、このモードに移行します。すべての割り込みをマスクする必要があります。		2	CPU 動作が停止します。SYSOSC はディセーブルのままです。LFOSC はイネーブルのままです。MCLK は LFCLK から実行されます。

表 3-9. STM8 デバイスと MSPM0 デバイスの動作モードの比較 (続き)

STM8		MSPM0		
動作モード	説明	動作モード		説明
アクティブ ホールト モード (STM8S)	CPU 動作が停止します。LSI または HSE 以外のすべての発振器がディセーブルになります。AWU 以外のほとんどすべてのペリフェラルが停止します。MVR レギュレータの電源はオンの状態になります。	STOP(2)	0	CPU 動作が停止します。SYSOSC のステータスが保持されます(1)。LFOSC はイネーブルのままです。ULPCLK は最大 4MHz に制限されます。PD0 はイネーブル、PD1 はディセーブルになります。また、ADC などのアナログ ペリフェラルは動作可能です。
			1	SYSOSC と ULPCLK が 4MHz に切り替えられます (STOP0 と同様)。
MVR 自動電源オフ対応のアクティブ ホールト モード(STM8S)	CPU 動作が停止します。LSI 以外の発振器がディセーブルになります。AWU 以外のほとんどすべてのペリフェラルが停止します。MVR レギュレータの電源はオフの状態になります。		2	CPU 動作が停止します。SYSOSC はディセーブルになります。ULPCLK は 32kHz で動作します。PD0 はイネーブル、PD1 はディセーブルになります。
アクティブ ホールト モード (STM8S ファミリ以外)	CPU 動作が停止します。LSI または LSE を除くすべての発振器がディセーブルになります。RTC、AWU などを除くほとんどすべてのペリフェラルが停止します。電圧レギュレータは低消費電力モードです。	STANDBY	0	CPU 動作が停止します。SYSOSC はディセーブルです。すべての PD0 ペリフェラルが ULPCLK と LFCLK を受信します。
ホールト モード	CPU 動作が停止します。発振器はディセーブルです(3)。ほとんどすべてのペリフェラルが停止します。電圧レギュレータは低消費電力モードです(3)。		1	STANDBY0 と同様に、TIMG0/1 のみが ULPCLK または LFCLK を受信します。
該当なし	該当なし	シャットダウン		利用可能なクロックがなくなり、デバイスはシャットダウンされます。

(1) RUN1 から STOP0 に遷移した場合 (SYSOSC がイネーブルで、MCLK は LFCLK から供給)、RUN1 のときと同様に、SYSOSC はイネーブルのままになります。RUN2 から STOP0 に遷移した場合 (SYSOSC がディセーブルで、MCLK は LFCLK から供給)、RUN2 のときと同様に、SYSOSC はディセーブルのままになります。
(2) MSPM0C デバイスには STOP1 モードはありません。
(3) STM8L001xx および STM8L101xx デバイスでは、IWDG が有効で「*ホールトモード中にウォッチドッグを無効にする*」オプションがディセーブルになっている場合、LSI 発振器は ホールト モード中も動作します。有効化されていて*ホールト モード中にウォッチドッグを無効にするオプション*がディセーブルで無効である場合、BEEP と IWDG のみがホールト モード中も動作を継続します。

STM8L05xx デバイスには、次の 5 つの低消費電力モードがあります:ウェイト モード、低消費電力実行モード、低消費電力ウェイト モード、アクティブ ホールト モード、ホールト モード。STM8L001xx および STM8L101xx デバイスには、次の 3 つの低消費電力モードがあります:ウェイト モード、アクティブ ホールト モード、ホールト モード。STM8 シリーズには、次の 4 つの低消費電力モードがあります:ウェイト モード、アクティブ ホールト モード、MVR 自動電源オフ対応のアクティブ ホールト、ホールト モード。

3.5.2 低消費電力モードでの MSPM0 機能

MSPM0 のペリフェラル モードは、低消費電力動作モードにおいては、利用可能な機能または動作速度を制限できます。具体的な詳細については、MSPM0 デバイス固有のデータシートに掲載されている「*動作モードでサポートされる機能*」の表を参照してください。関連する 2 つのデータシートを以下の例に示します:

- *MSPM0L134x、MSPM0L130x ミックスド シグナル マイクロコントローラ データシート*
- *MSPM0C110x、MSPS003 ミックスド シグナル マイクロコントローラ データシート*

MSPM0 デバイスの追加機能は、一部のペリフェラルが非同期高速クロック要求を実行できることです。これにより、MSPM0 デバイスを、ペリフェラルがアクティブでない低消費電力モードに移行しつつ、ペリフェラルをトリガまたはアクティブにすることもできます。非同期高速クロック要求が発生した場合、MSPM0 デバイスは内部発振器を高速にすばやく立ち上げたり、一時的に高い動作モードに移行して差し迫った動作を処理したりすることができます。これにより、最小消費電力モードでのスリープ中に、タイマ、コンパレータ、GPIO からの CPU の高速ウェークアップ、SPI、UART、I2C の受信、または DMA 転送と ADC 変換のトリガを行うことができます。非同期クロック要求の実装とペリフェラルのサポートおよび目的の具体的な詳細については、MSPM0 のデバイス固有の TRM の対応する章を参照してください。

- *『MSPM0 L シリーズ 32MHz マイクロコントローラ テクニカル リファレンス マニュアル』*
- *『MSPM0 C シリーズ 24MHz マイクロコントローラ テクニカル リファレンス マニュアル』*

3.5.3 低消費電力モードへの移行

MSPM0 デバイスは、イベントの待機、**_WFE( )**、または割り込みの待機、**_WFI( )**、の命令を実行するときに、低消費電力モードに移行します。低消費電力モードは、現在の電力ポリシー設定によって決定されます。デバイスの電力ポリシーは、ドライバ ライブラリ関数によって設定されます。次の関数呼び出しは、電力ポリシーをスタンバイ 0 に設定します。

DL_SYSCTL_setPowerPolicySTANDBY0 ( );

STANDBY0 は、任意の動作モードに置き換えることができます。電源ポリシーを管理する driverlib API の全リストについては、『MSPM0 SDK API ガイド』の該当するセクションを参照してください。また、さまざまな動作モードの開始方法を示す以下のサンプル コードも参照してください。すべての MSPM0 デバイスで、同様のサンプルを利用できます。

3.5.4 低消費電力モードのサンプル コード

SDK のインストール先に移動し、低消費電力モードのサンプル コードを「examples > nortos > LP name > driverlib」から検索します。

3.6 割り込みとイベントの比較

3.6.1 割り込みと例外

MSPM0 と STM8 は両方とも、デバイスで利用可能なペリフェラルに応じて、割り込みベクトルと例外ベクトルを登録し、マップします。各デバイス ファミリの割り込みベクトルの概要と比較を表 3-10 に示します。

表 3-10. 割り込みの比較

特長	STM8L、STM8S	MSPM0L、MSPM0C および MSPM0H
割り込みタイプ	ペリフェラル割り込み：使用するデバイスによって決まります	ペリフェラル割り込み：MSPM0L の NVIC は最大 19 個のペリフェラル割り込みベクタをサポート MSPM0C の NVIC は最大 23 個のペリフェラル割り込みベクタをサポート MSPM0H は最大 22 個のペリフェラル割り込みベクタをサポート(2)
	外部割り込み：STM8L バリュー ラインは 11 個のベクタ STM8L101x は 10 個のベクタ STM8S は 5 個のベクタ(1)	
	ノンマスカブル割り込み：RESET、TRAP (ソフトウェア割り込み)、TLI (トップレベル ハードウェア割り込み)(3)	Reset、Hard Fault、SVCall、PendSV、SysTick
		NMI：ソフトウェアトリガ、SYSCTL からのハードウェアエラー信号
優先レベル	ハードウェア優先度レベル：割り込みマッピングの IRQ 番号	デフォルトの優先度レベル：NVIC 番号(4)
	ノンマスカブル割り込みは、ソフトウェアの優先度が最も高いと見なされます	システム例外 (リセット、NMI、ハードフォルト) にはそれぞれ固定の優先レベル (-3、-2、-1) があります
	マスカブル割り込みには、次の 4 つのソフトウェア優先度レベルがあります：0 (メイン)、1、2、3 (ソフトウェアによる優先度設定が無効)	ペリフェラル割り込みには、以下の 4 つのプログラム可能な優先レベルがあります：0、64、128、192
優先度セット	ITC_SPRx レジスタ：各割り込みベクタに対してソフトウェア優先度を定義するために使用します(5) CCR レジスタ：現在発生している割り込み要求のソフトウェア優先度を自動的にロードするために使用します(6)	NVIC の IPRx レジスタ：ペリフェラル割り込み優先度レベルを設定するために使用します
割り込みマスク	対応する割り込み許可ビットは、各ペリフェラルの制御レジスタで設定します	ペリフェラル側の IMASK レジスタ：イベントに伝搬する割り込み条件を設定するために使用します(7)
		NVIC の ISER および ICER レジスタ：ペリフェラル割り込みをイネーブルまたはディセーブルにするために使用します

(1) 外部割り込みを生成するには、対応する GPIO ポートを入力モードかつ割り込みイネーブルに設定する必要があります。
(2) NVIC に加え、MSPM0 デバイスには INT_GROUP0 や INT_GROUP1 などの割り込みグループ モジュールが搭載されている場合があり、より多くのペリフェラル割り込みを NVIC へ接続することが可能です。外部割り込み / GPIO 割り込みは INT_Group1 モジュールにあります。
(3) STM8S シリーズのみがトップレベル ハードウェア割り込み (TLI) をサポートしています。
(4) NVIC 番号は、複数の NVIC 割り込みに同じプログラム可能な優先度がある場合の相対的な割り込み優先度を示します。
(5) VECTxSPR[1:0] に 10 (優先レベル 0) を書き込むことは禁止されています。10 を書き込んでも、前の値が保持され、割り込みの優先度は変更されません。
(6) ノンマスカブル割り込みソースは、CCR レジスタのビット I1 と I0 の状態に関係なく処理されます。
(7) MSPM0 のイベントハンドラに、MSPM0 のイベント ハンドラと関連する管理レジスタを示します。

MSPM0 デバイスの場合、割り込みまたは例外の優先度の値が低いほど、優先度の高い値を持つ割り込みよりも優先度が高くなります。プロセッサが割り込みを処理中の場合、より高い優先度の割り込みのみがその処理をプリエンプトすることができます。STM8 デバイスの場合、割り込みまたは例外の優先度の値が高いほど、優先度の低い値を持つ割り込みよりも優先度が高くなります。また、STM8 デバイスには、「コンカレント モード」と「ネスト モード」の 2 つの割り込み管理モードがあります。詳細は、デバイス固有のデータシートを参照してください。

3.6.1.1 MSPM0 の割り込み管理

MSPM0 デバイスでは、NVIC の IPRx レジスタで各ペリフェラル割り込みソースの優先レベルを設定し、NVIC の ISER および ICER レジスタでペリフェラル割り込みソースをマスク/マスク解除します。各ペリフェラル割り込みには、各種の割り込み条件があります。たとえば、ペリフェラル割り込みソースとして、UARTx は送信割り込みや受信割り込みなど複数の割り込み条件を持っています。これらの割り込み条件は、ペリフェラル側の 6 つの標準レジスタで管理されています。図 3-2 に、ペリフェラル割り込みの階層を示します。

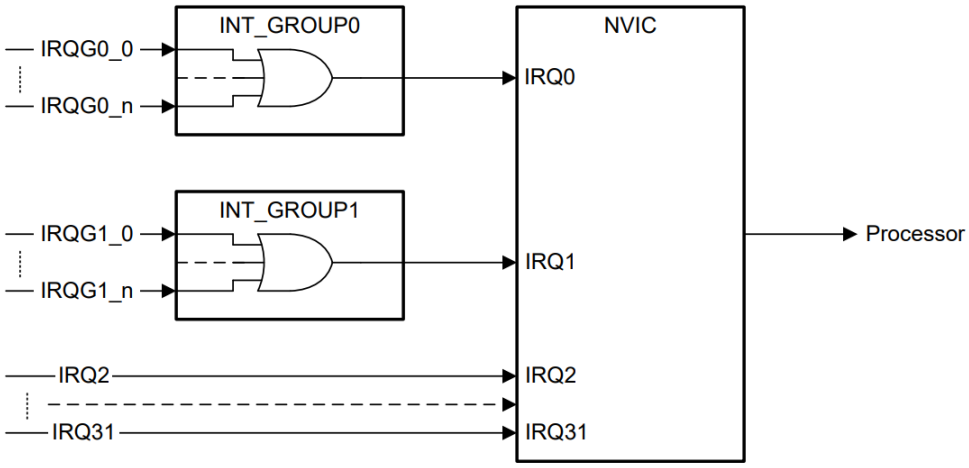


図 3-2. MSPM0 のペリフェラル割り込み階層

3.6.1.2 STM8 の割り込みコントローラ (ITC)

図 3-3 に、STM8 の割り込み処理のフローチャートを示します。割り込みサービスルーチン (ISR) 内で SIM 命令を使用して割り込みマスクビット I0 と I1 を設定されている場合、RIM 命令で割り込みマスクを削除すると、ソフトウェア プライオリティがレベル 0 に設定されます。割り込みサービスルーチンは必ず IRET 命令で終了してスタックに保存されたレジスタの内容を戻す必要があります。この IRET 命令により、スタックからビット I1 と I0 が復元され、プログラムの実行が再開されます。

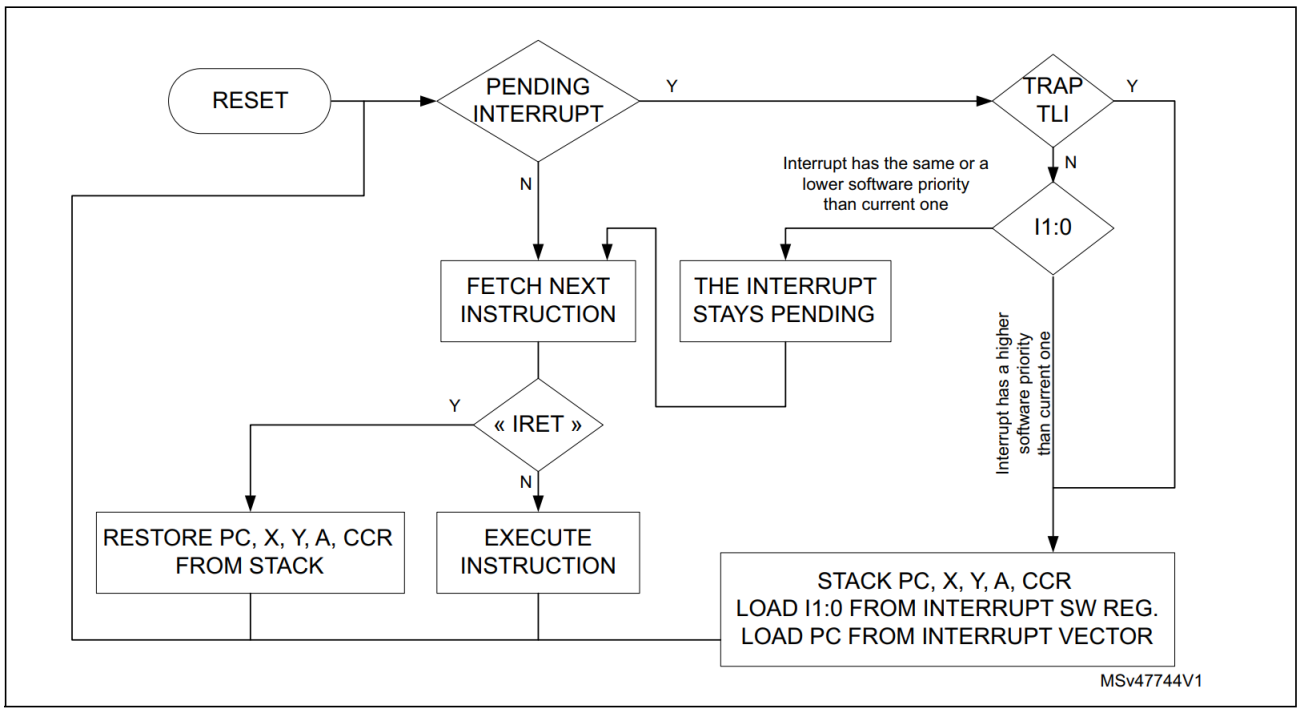


図 3-3. 割り込み処理のフローチャート

3.6.2 MSPM0 のイベントハンドラ

MSPM0 マイコンには、ある機能から別の機能へデジタルイベントを送るイベント マネージャーが搭載されています。イベント マネージャは、静的なルートとプログラマブルなルートの組み合わせを含むイベント ファブリックによって相互接続された一連の定義済みイベント パブリッシャ (ジェネレータ) およびサブスクライバ (レシーバ) によるイベント転送を実装しています。また、イベント マネージャはパワー マネージメントおよびクロック ユニット (PMCU) とのハンドシェイクを実行し、トリガされたイベント アクションを実行するために必要なクロックと電力ドメインが存在することを確認することもできます。

イベント マネージャによって転送されるイベントには、以下が含まれます。

- 割り込み要求 (IRQ) として CPU に転送されるペリフェラル イベント
- DMA トリガとして DMA に転送されるペリフェラル イベント
- ハードウェアでの動作を直接トリガするため、別のペリフェラルに転送されるペリフェラル イベント

イベント マネージャは、イベントファブリックを介してイベント パブリッシャをイベント サブスクライバに接続します。イベント ファブリックには、次の 3 種類があります：CPU 割り込み (固定イベントルート)、DMA ルート、汎用ルート。例として、図 3-4 に汎用ルートを示します。

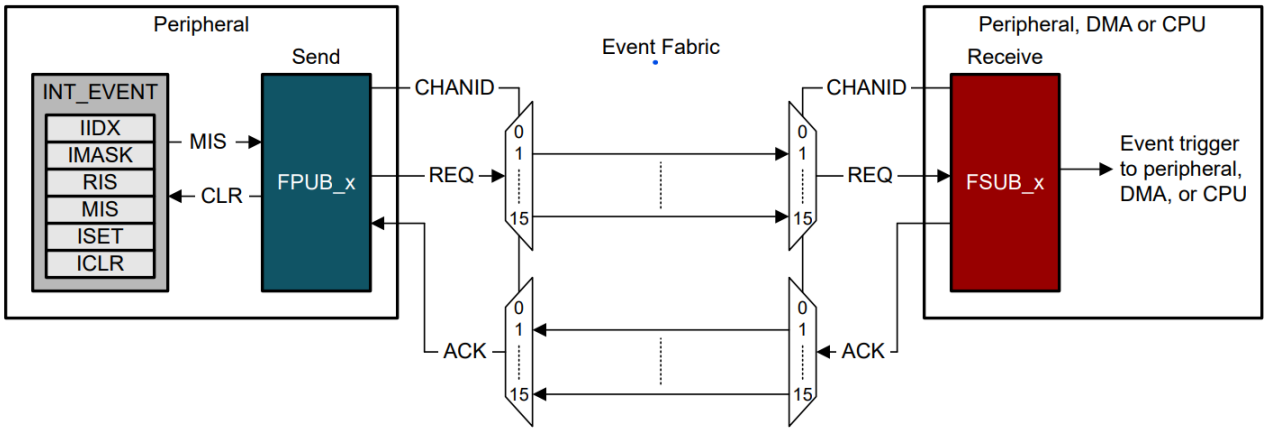


図 3-4. 汎用イベント ルート

イベント マネージャのレジスタ セットには、次の 6 つの標準的なレジスタがあります:RIS、IMASK、MIS、ISET、ICLR、IIDX。イベント レジスタは相互接続されています (図 3-5 を参照)。マスクが解除されると、保留中の割り込みが RIS レジスタと MIS レジスタの両方に表示され、イベントが生成されます。CPU 割り込みイベントルートによる CPU 割り込みの場合、IIDX レジスタの読み出しにより、RIS レジスタと MIS レジスタの優先度が最も高い保留中の割り込みがクリアされ、その割り込みのインデックスがアプリケーションソフトウェアに返されます。

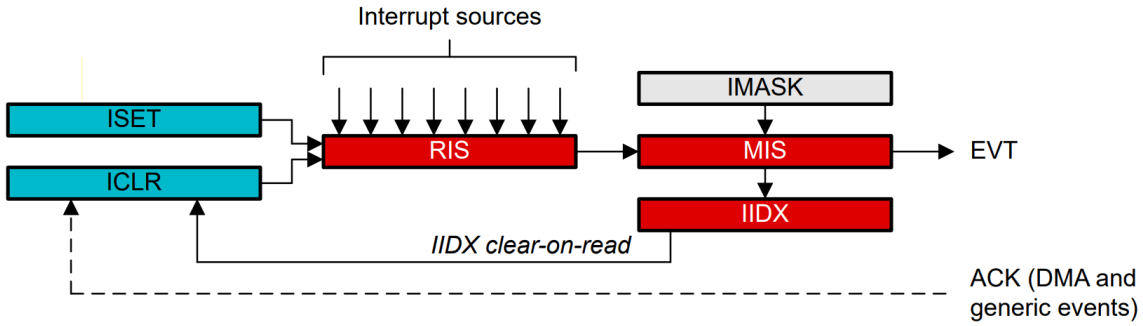


図 3-5. イベント管理の登録関係

図 3-6 にイベント マップを示します。さまざまなイベント遷移を実現するために、さまざまなペリフェラルがさまざまなイベントファブリックを介してルーティングされます。MSPM0 での詳しいイベント ハンドラの使用方法については、MSPM0 L シリーズ 32Mhz マイクロコントローラ テクニカル リファレンス マニュアルまたは MSPM0 C シリーズ 24Mhz マイクロコントローラ テクニカル リファレンス マニュアルのイベントセクションを参照してください。

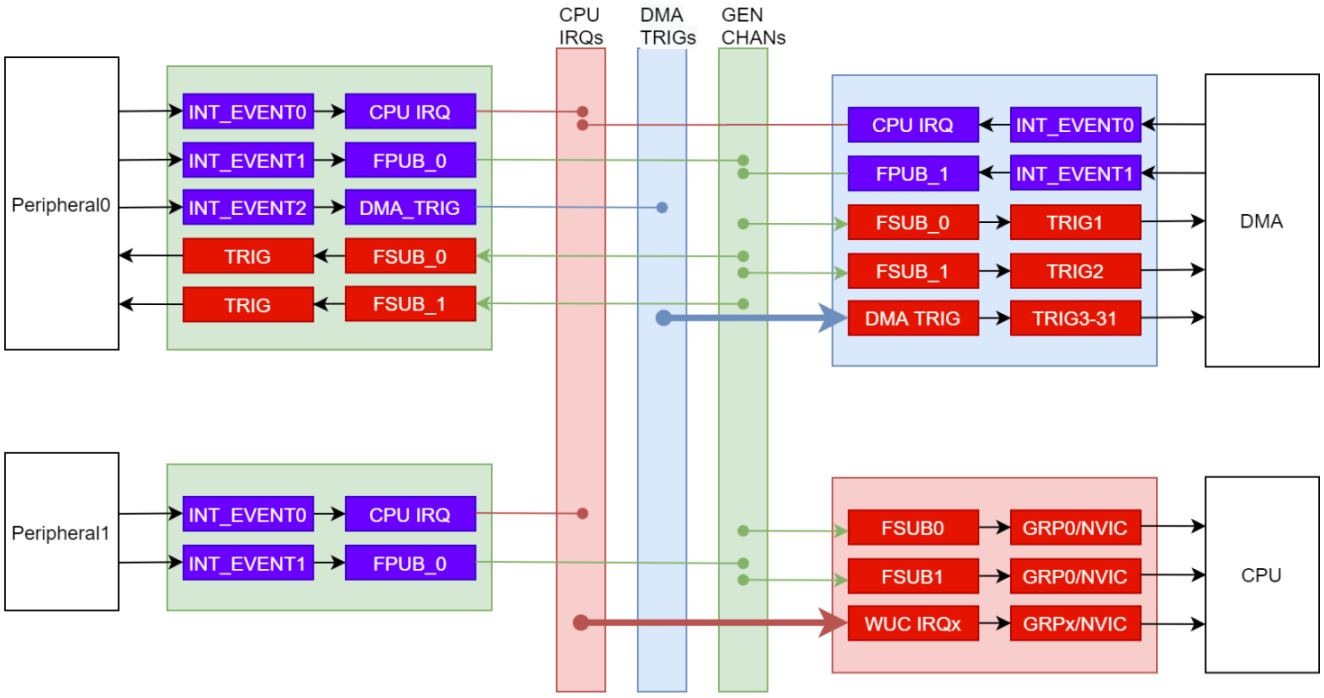


図 3-6. MSPM0 イベントおよび割り込み処理

3.6.3 イベント管理の比較

MSPM0 と STM8 では、イベント管理の構造や機能が異なります。MSPM0 MCU は、イベント ハンドラを使用してさまざまなイベントを管理します。STM8 MCU は、異なるコントローラを使用してさまざまなイベントを管理します。MSPM0 のイベントハンドラの比較と STM8 のイベント管理を表 3-11 に示します。

表 3-11. イベント管理の比較

特長		STM8L、STM8S	MSPM0L、MSPM0C および MSPM0H
パブリッシャ		ペリフェラル	ペリフェラル
サブスクライバ		CPU、DMA トリガ、ペリフェラル	CPU、DMA トリガ、ペリフェラル
管理スタイル	ペリフェラル → CPU	割り込みコントローラ (ITC)	イベント マネージャ
	ペリフェラル → DMA	DMA コントローラ(1)	
	ペリフェラル → ペリフェラル	トリガコントローラ(2)	
ルート タイプ		ポイント ツー ポイント (1:1)	ポイント ツー ポイント
			ポイントツーツーツー (スプリッタ)(3)

(1) すべての STM8 デバイスに DMA コントローラーが搭載されているわけではありません。STM8L001xx ファミリ、STM8L101xx ファミリ、STM8S シリーズは DMA 機能をサポートしていません。
(2) TIMx のトリガーコントローラは、トリガ出力信号 (TRGO) を生成して、他の TIM タイマや ADC、DAC を起動することができます。
(3) 汎用ルートチャネルは、選択されているチャネルに応じて、1 つのサブスクライバ (1:1) または 2 つのサブスクライバ (1:2 スプリッタルート) に設定できます。

3.7 デバッグとプログラミングの比較

3.7.1 デバッグモードの比較

Arm SWD 2 線式 JTAG ポートは、MSPM0 デバイスのメインのデバッグおよびプログラミング インターフェイスです。このインターフェイスは通常、アプリケーション開発時および量産プログラミング時に使用されます。

MSPM0 デバイスとは異なり、STM8 デバイスには SWD 2 線式 JTAG ポートがありません。STM8 デバイスの主なデバッグおよびプログラミング インターフェイスは、超高速メモリ プログラミングが可能な 1 線式ハードウェア インターフェイス (SWIM) です。SWIM ピンは、デバッグにも使いたい場合は制限がありますが、標準の I/O としても使用できます。最も安全な方法は、PCB にストラップ オプションを設けることです。

3.7.2 プログラミングモードの比較

ブートストラップ ローダ (BSL) プログラミング インターフェイスは、ARM SWD および STM8 SWIM に対する代替プログラミング インターフェイスです。このインターフェイスはプログラミング機能のみを提供し、通常は標準の組込み通信インターフェイスを通して使用します。これにより、システム内または外部ポート内の他の組み込みデバイスへの既存の接続を経由して、ファームウェアを更新できます。プログラミングの更新はこのインターフェイスの主な目的ですが、BSL は初期の量産プログラミングにも利用できます。

3.7.2.1 ブートストラップ ローダ (BSL) のプログラミング オプション

MSPM0 と STM8 の両デバイスは、BSL プログラミング インターフェイスをサポートしています。表 3-12 に MSPM0 と STM8 の各デバイス ファミリのさまざまなオプションと機能の比較を示します。

表 3-12. BSL 機能の比較

BSL の機能	STM8L、STM8S	MSPM0L	MSPM0C	MSPM0H
組み込み BSL コード ストレージ	ROM (1)	ROM (2)	非対応	非対応
カスタム化可能	いいえ	構成可能な起動ピン、COMM ピン、プラグイン機能	いいえ	いいえ
セカンダリ BSL コード ストレージ	UBC プログラム エリア(1)	メイン フラッシュ(2)	メイン フラッシュ	メイン フラッシュ
BSL はブランク デバイスで開始されました	あり	あり	該当なし	該当なし
プログラミング インターフェイスの自動検出	あり	あり	該当なし	該当なし
セキュリティ	読み出し保護 (ROP)、コマンド チェックサム	セキュア ブート オプション、CRC 保護 キーストア付き AES256、TRNG	CRC、ファイアウォール、IP 保護	CRC、ファイアウォール、IP 保護
メソッドを起動する	BSL オプションバイトが 0x55AA であるか、またはプログラム メモリが未使用であるかを確認します	BOOTRST で起動ピンが High になる、SW エントリ	BOOTRST で起動ピンが High になる、SW エントリ	BOOTRST で起動ピンが High になる、SW エントリ
サポートされているインターフェイス				
UART	あり	あり	セカンダリ BSL	セカンダリ BSL
I2C	非対応	あり	セカンダリ BSL	セカンダリ BSL
SPI	あり	カスタム プラグインが必要	セカンダリ BSL	セカンダリ BSL
CAN	あり (3)	プラグインを計画中 (4)	セカンダリ BSL	セカンダリ BSL

(1) STM8Sx03xx、STM8S001xx、STM8L101xx、STM8L001xx の各デバイスには、BSL が内蔵されていません (マイクロコントローラに ROM BSL は実装されていません)。これらのデバイスを使用する場合、ユーザー自身が BSL コードを作成し、そのコードを UBC プログラム領域に保存する必要があります。
(2) MSPM0C デバイスには ROM BSL コードはありません。これらのデバイスを使用する場合、ユーザー自身がカスタマイズされたプラグイン インターフェイスと BSL コアを含む BSL コードを作成し、そのコードをメイン フラッシュ メモリに保存する必要があります。
(3) STM8 デバイスの CAN ペリフェラルは、外部クロック (8MHz、16MHz、または 24MHz) が存在する場合にのみ使用できます。
(4) 一部のデバイスのみ。

4 デジタル ペリフェラルの比較

4.1 汎用 I/O (GPIO、IOMUX)

MSPM0 GPIO 機能は、STM8S および STML シリーズが提供するほぼすべての機能を網羅しています。STM8 では、ピン機能やポート機能という用語を使用します。この用語は、デバイスのピン管理や割り込みの生成など、ピンに関わるあらゆる機能を包括的に指します。MSPM0 GPIO および IOMUX の機能について以下に説明します:

- MSPM0 GPIO とは、IO の読み取りと書き込み、割り込みの生成などを実行できるハードウェアのことです。
- MSPM0 IOMUX とは、さまざまな内部デジタル ペリフェラルをピンに接続するために使用するハードウェアのことです。IOMUX は、GPIO など、多くの異なるデジタル ペリフェラルにサービスを提供しますが、これらに限定されるものではありません。

MSPM0 GPIO と IOMUX を組み合わせることで、STM8 GPIO と同じ機能を実現できます。さらに、MSPM0 は、DMA 接続、制御可能な入力フィルタリング、イベント機能など、STM8L および STM8S シリーズのデバイスでは利用できない機能を提供します。

表 4-1. GPIO 機能の比較

機能	STM8S、STM8L	MSPM0L、MSPM0L、MSPM0H
出力モード	プッシュプル オープンドレイン	プッシュプル プルダウン付きのオープンドレイン Hi-Z
入力モード	プルアップ フローティング アナログ	フローティング プルアップまたはプルダウン アナログ
GPIO 速度の選択	各 I/O の速度選択 Speed0 最大 2MHz Speed 最大 10MHz	MSPM0 では、すべての I/O ピンで標準 I/O (SDIO) を提供しています。 MSPM0 高速 IO (HSIO) は、選択されたピンで利用可能です。 SDIO と HSIO はどちらも VDD≥2.7V で最大 32 MHz、HSIO は VDD≥1.71V で最大 24MHz で動作します
アトミック ビットのセットとリセット	あり	あり
代替機能	ODR、IDR、および DDR レジスタを使用	IOMUX を使用
高速トグル	少なくとも 2 クロックごと	クロックサイクルごとにピンを切り替え
ウェークアップ	外部割り込み	GPIO ピンの状態変更
GPIO は DMA によって制御される	いいえ	*MSPM0 でのみ利用可能*
ユーザー制御の入力フィルタリングにより、1、3、8 ULPCLK 周期未満のグリッチを除去する	いいえ	*MSPM0 でのみ利用可能*
ユーザー制御可能な入力ヒステリシス	いいえ	*MSPM0 でのみ利用可能*

GPIO サンプルコード:GPIO サンプル コードの詳細については、『MSPM0 SDK サンプル ガイド』を参照してください。

4.2 UART (Universal Asynchronous Receiver-Transmitter)

STM8S シリーズと MSPM0 はどちらも非同期 (クロックレス) 通信を実行するためのペリフェラルを備えています。STM8L シリーズは USART (汎用同期・非同期送受信機能) を備えており、外部機器との全二重データ通信が可能です。

表 4-2. UART 標準機能の比較

機能	STM8S および STML	MSPM0L、MSPM0L、MSPM0H
データの方向	LSB 先頭	MSB 先頭または LSB 先頭
ハードウェア フロー制御	フロー	あり
設定可能なストップ ビット	1、2	1、2
Tx、Rx FIFO の深度	2	4
マルチプロセッサ	あり	あり
低消費電力モードでの動作	あり (ウェイト モードのみ)	あり (すべての低消費電力)
1 線式半二重通信	あり	あり(1)
低消費電力モードからのウェークアップ	あり	あり
パリティ	偶数、奇数	偶数、奇数
DMA を使用した通信	あり(2)	あり

(1) 伝送と受信の間でペリフェラルを再構成する必要があります。
(2) STM8L バリュー ラインのみに DMA が搭載されています。

表 4-3. UART の高度な機能の比較

機能	STMS と STML	MSPM0L、MSPM0L、MSPM0H
同期モード	あり(1)	いいえ
データ長	8、9	5、6、7、8
オーバーサンプリング	いいえ	16x 8x 3x
LIN ハードウェアのサポート	あり(2)	あり
DALI ハードウェアのサポート	いいえ	あり
IrDA ハードウェア サポート	あり	あり
マンチェスターコード HW サポート	いいえ	あり
スマート カード モード (ISO7816)	あり	あり
外部ドライバ イネーブル	いいえ	あり
グリッチ フィルタ	いいえ	あり

(1) STM8S の場合、UART1、UART2、UART4 のみが同期通信用の伝送クロック出力を備えています。STM8L には、同期通信に対応した UASART モジュールが搭載されています。
(2) STM8 LIN モジュールのハードウェア モジュールは、MSPM0 よりも次のようなより包括的な機能を備えています:LIN ブレークやデリミタの生成、ヘッダーエラーの検出など。

UART サンプルコード:UART サンプル コードの詳細については、『MSPM0 SDK サンプル ガイド』を参照してください。

4.3 シリアル・ペリフェラル・インターフェイス (SPI)

MSPM0 と STM8 はどちらも、シリアル ペリフェラル インターフェイス (SPI) をサポートしています。全体として、MSPM0 および STM8 SPI のサポート内容はおおむね同等ですが、違いについて表 4-4 に示します。

表 4-4. SPI 機能の比較

機能	STM8S、STM8L	MSPM0L、MSPM0L および MSPM0H
操作ワイヤ	SCK、MOSI、MISO、NSS	SCLK、PICO、POCI、CSx
コントローラまたはペリフェラルの動作	あり	あり
マルチ コントローラ機能	あり	いいえ
データの順序	MSB 先頭または LSB 先頭	MSB 先頭または LSB 先頭
データ ビット幅 (コントローラ モード)	言及していません	4～16 ビット
データ ビット幅 (ペリフェラル モード)		7～16 ビット
最大速度	10MHz	16MHz
シンプレックス通信 (単方向データ ライン)	あり	あり
ハードウェア チップ セレクト マネージメント	あり (1 つのペリフェラル)	あり (4 つのペリフェラル)
I/O クロックの位相制御	あり	あり
SPI フォーマットのサポート	Motorola	Motorola、テキサス・インスツルメンツ、MICROWIRE
ハードウェア CRC	あり (STM8S)	なし、MSPM0 は SPI パリティ モードを備えています
低消費電力モード	ウェイト モード	スリープ モード
TX FIFO の深さ	1 (バッファ)	4
RX FIFO の深さ	1 (バッファ)	4

SPI サンプルコード：SPI サンプル コードの詳細については、『MSPM0 SDK サンプル ガイド』を参照してください。

4.4 Interintegrated Circuit Interface (I2C)

MSPM0 と STM8 はどちらも I2C ペリフェラルをサポートしています。全体として、MSPM0 および STM8 I2C のサポート内容はおおむね同等ですが、重要な違いについて表 4-5 に示します。

表 4-5. I2C 機能の比較

機能	STM8S、STM8L	MSPM0L、MSPM0C および MSPM0H
コントローラ モードとターゲット モード	あり	あり
マルチ コントローラの機能	あり	あり
最大データ転送レート	400 kHz (高速)	1Mbps (高速モード プラス)
アドレッシング モード	7 ビットまたは 10 ビット	7 ビット
アドレス番号 (ターゲット モード)	2 つのアドレス	2 つのアドレス
ゼネラル コール	あり	あり
イベント管理	あり	あり
PEC 管理	あり	あり
クロック ストレッチ	あり	あり
ウェークアップ機能 (低消費電力モード)	あり	あり
ソフトウェア・リセット	あり	あり
FIFO、バッファ	1 バイト	TX：8 バイト
		RX：8 バイト
DMA	あり	あり
プログラム可能なアナログおよびデジタル ノイズ フィルタ	あり	あり

I2C サンプルコード：I2C サンプル コードの詳細については、『MSPM0 SDK サンプル ガイド』を参照してください。

4.5 タイマ (TIMGx、TIMAx)

STM8 と MSPM0 はどちらも、さまざまなタイマを搭載しています。MSPM0 は低消費電力監視から高度なモーター制御までのユースケースをサポートする、さまざまな機能を備えたタイマを提供しています。

表 4-6. タイマの命名

STM8L		STM8S		MSPM0L		MSPM0C		MSPM0H	
タイマ名	略称	タイマ名	略称	タイマ名	略称	タイマ名	略称	タイマ名	略称
		高度な制御	TIM1	高度な制御	TIMA0 (MSPM0Lx22x のみ)	高度な制御	TIMA0	高度な制御	TIMA0
汎用	TIM2、3	汎用	TIM2、3.5	汎用	TIMG0、1、2、4、5、8、12	汎用	TIMG1、2、8、14	汎用	TIMG1、2、8、14
基本	TIM4	基本	TIM4、6						

表 4-7. タイマ機能の比較

機能	STM8L	STM8S	MSPM0L、MSPM0C および MSPM0H
分解能	8、16 bit	8、16 bit	16 ビット
PWM	あり	あり	あり
キャプチャ	あり	あり	あり
比較	あり	あり	あり
繰り返しカウンタ	いいえ	あり	あり
ワンショット	あり	あり	あり
アップ ダウン カウント機能	あり	あり	あり
低消費電力モード	あり	あり	あり
QEI のサポート	いいえ	いいえ	あり
プログラマブル プリスケーラ	あり	あり	あり
シャドウ レジスタ モード	あり	あり	あり
イベント、割り込み	あり	あり	あり
自動リロード機能	あり	あり	あり
フォルト処理	あり	あり	あり

表 4-8. タイマ モジュールの代替品

STM8	MSPM0 同等品	推論
TIM1	TIMA0	高度な制御、16 ビットの分解能、アップ/ダウン カウンタ、リピート カウンタ
TIM2、3、5	TIMG0-11、TIMG14	16 ビットの分解能、汎用、キャプチャ / 比較機能
TIM4、6 年	任意(1)	ベーシック タイマ

表 4-9. タイマの使用事例の比較

機能	STM8L、STM8S	MSPM0L、MSPM0C および MSPM0H
PWM	TIM1TIM2、3、5	すべてのタイマ
キャプチャ	TIM1TIM2、3、5	すべてのタイマ
比較	TIM1TIM2、3、5	すべてのタイマ
ワンショット	TIM1TIM2、3、5	すべてのタイマ
プリスケーラ	すべてのタイマ	すべてのタイマ
同期	すべてのタイマ	すべてのタイマ

タイマ サンプルコード:タイマ サンプル コードの詳細については、『MSPM0 SDK サンプル ガイド』を参照してください。

4.6 ウィンドウ付きウォッチドッグ タイマ (WWDT)

STM8 と MSPM0 はどちらもウィンドウ ウォッチドッグ タイマを備えています。ウィンドウ ウォッチドッグ タイマ (WWDT) は、指定された時間内にアプリケーションがチェックインに失敗した場合、システム リセットを開始します。

表 4-10. WWDT の命名法

STM8	MSPM0
独立したウォッチドッグ (IWDG) ウィンドウ付きウォッチドッグ (WWDG)(1)	ウィンドウ付きウォッチドッグ タイマ (WWDT)

(1) WWDG は STM8L のみが備えています。

表 4-11. WDT 機能の比較

機能	STM8L、STM8S	MSPM0L、MSPM0C および MSPM0H
ウィンドウ モード	あり	あり
インターバル タイマ モード	いいえ	あり
LFCLK 電源	あり	あり
割り込み	あり	あり
カウンタの分解能	8 ビット (IWDG) 7 ビット (WWDG)	25 ビット
クロック デバイダ	あり	あり

WWDT サンプルコード:WWDT サンプル コードの詳細については、『MSPM0 SDK サンプル ガイド』を参照してください。

5 アナログ ペリフェラルの比較

5.1 A/D コンバータ (ADC)

STM8 と MSPM0 はどちらも、アナログ信号を同等のデジタル信号に変換するための ADC ペリフェラルを搭載しています。STM8 の場合、STM8L001XX および STM8L101XX には ADC モジュールが搭載されていません。STM8S シリーズには 10 ビット ADC が、その他の STM8L シリーズには 12 ビット ADC が搭載されています。MSPM0 の場合、どちらのデバイス シリーズも、12 ビット ADC を搭載しています。表 5-1 と 表 5-2 に、ADC のさまざまな機能とモードの比較を示します。

表 5-1. 機能セットの比較

機能	STM8S	STM8L	MSPM0
分解能 (ビット)	10	12	12、10、8
変換レート	0.43 Msps	1Msps	1.68 Msps
ハードウェア平均化	いいえ	いいえ	あり
FIFO	いいえ	いいえ	あり
ADC 基準電圧 (V)	内部：1.224	内部：1.48、VDD	内部：1.4、2.5、VDD
	外部： 2.75V ≤ V_{REF} ≤ VDDA	外部：2.4V ≤ V_{REF} ≤ VDDA	外部： 1.4 ≤ V_{REF} ≤ V_{DD} (1)
動作時電力モード	待機電力と低消費電力待機	ウェイト	動作、スリープ、停止、スタンバイ(2)
自動パワー ダウン	いいえ	いいえ	あり
外部入力チャネル	最大 16	最大 28	MSPM0L：最大 16、MSPM0C：最大 27、MSPM0H：最大 27
内部入力チャネル	温度センサ 内部リファレンス電圧	温度センサ 内部リファレンス電圧	温度センサ 電源監視、アナログ シグナル チェーン
DMA のサポート	いいえ	あり	あり
ADC ウィンドウ コンパレータ ユニット	いいえ	いいえ	あり
ADC の数	最大 1	最大 1	最大 1

(1) 外部リファレンスをサポートしているのは、C シリーズの MSPM0C1105 と MSPM0C1106 のみです。
(2) ADC はスタンバイ モードでトリガできるため、動作モードが変化します。

表 5-2. 変換モード

STM8	MSPM0	備考
シングル変換モード	シングル チャネル シングル変換	ADC は、1 つのチャネルを 1 回サンプリングして変換します
シングル スキャン モード	チャネル変換のシーケンス	ADC は一連のチャネルをサンプリングし、1 回変換します。
連続モードとバッファ連続モード	シングル チャネル変換を繰り返します	選択した 1 つのチャネルが繰り返しサンプリング、変換されます
連続スキャン モード	チャネル変換のシーケンスを繰り返します	チャネルのグループが繰り返しサンプリング、変換されます

ADC サンプルコード：ADC サンプル コードの詳細については、『MSPM0 SDK サンプル ガイド』を参照してください。

5.2 コンパレータ (COMP)

STM8 の場合、STML バリュー ラインのみコンパレータを搭載しており、S シリーズとその他の L シリーズは搭載していません。MSPM0 では、L シリーズにはオプションのペリフェラルとして内蔵コンパレータが用意されていますが、C シリーズにはありません。どちらのファミリのデバイスでも、これらのコンパレータ COMPx と呼ばれます。ここで、「*x*」の最後の文字は、検討対象のコンパレータ モジュールを表します。コンパレータ モジュールは、さまざまな内部および外部ソースを入力として受け取ることができ、電源モードの切り替えや PWM 信号の制御などに使用できます。MSPM0 および STM8 コンパレータ モジュールの機能別の比較については 表 5-3 を参照してください。

表 5-3. COMP 機能セットの比較

機能	STM8L バリュー ライン	MSPM0L および MSPM0C[3]
使用可能なコンパレータ	最大 2	最大 1
出力のルーティング	外部 I/O	多重化された I/O ピン
	TIM1 BRK または OCREFCLR 入力 TIM2/TIM3 入力キャプチャ 2	-
	割り込みまたはイベント インターフェイス	割り込みまたはイベント インターフェイス
正入力	外部 I/O	多重化された I/O ピン OPA1 出力 DAC8[1] 出力
負入力	内部基準電圧 内部基準電圧の分数値 (1/4、1/2、3/4) DAC 出力 3 つの外部 I/O のいずれか	多重化された I/O ピン 内部温度センサ OPA0 出力 DAC8 出力
プログラマブル ヒステリシス	いいえ	なし、10mV、20mV、30mV
		DAC8 を使用するその他の値は、0V～V_{DD} です
レジスタ ロック	いいえ	あり、COMP レジスタの一部 (書き込みにはキーが必要)
ウィンドウ コンパレータの構成	あり	なし (シングル COMP)
入力短絡モード	いいえ	あり
動作モード	最適速度、消費率	高速、低消費電力
出力フィルタリング	いいえ	ブランキング フィルタ
		調整可能なアナログ フィルタ
出力極性の制御	いいえ	あり
割り込み	立ち上がりエッジ	立ち上がりエッジ
	立ち下がりエッジ	立ち下がりエッジ
	両方のエッジ	出力準備完了
入力交換 (2)	いいえ	あり

(1) 8 ビット DAC は COMP に内蔵されています。
(2) 入力切り替えモードを有効にすると、コンパレータの正入力端子と負入力端子の信号が入れ替わります。さらに、コンパレータからの出力信号も反転されます。
(3) MSPM0H には現在 COMP は搭載されていません。

COMP サンプルコード：COMP サンプル コードの詳細については、『MSPM0 SDK サンプル ガイド』を参照してください。

5.3 基準電圧 (VREF)

STM8 と MSPM0 はどちらも内部基準電圧を備えており、内部ペリフェラルに基準電圧を供給したり、内部基準電圧を外部ペリフェラルに出力したりするために使用できます。STM8 では、内部基準電圧を備えているのは STM8L のバリューラインのみです。

表 5-4. VREF 機能セットの比較

機能	STM8S	STM8L	MSPM0
内部基準電圧 (V)	1.8、1.2	1.22	1.4、2.5
外部基準電圧 (V)	2.4 ≤ V_{REFP} ≤V_{DDA} V	2.4 ≤ V_{REFP} ≤V_{DDA} V	1.4 ≤ V_{REF} ≤ V_{DD} V(1)
内部基準電圧の出力	いいえ	あり	あり
ADC への内部接続	あり	あり	あり
DAC への内部接続	該当なし	あり	該当なし
COMP への内部接続	該当なし	あり	あり

表 5-4. VREF 機能セットの比較 (続き)

機能	STM8S	STM8L	MSPM0
OPA への内部接続	該当なし	該当なし	MSPM0L はサポートされています。MSPM0C および H はサポートされていません

(1) MSPM0C1103 と MSPM0C1104 は、外部参照電圧をサポートしていません。

MSPM0 VREF の場合、パワー ビット、PWREN Bit0 (ENABLE) をイネーブルにする必要があります。

VREF サンプルコード:VREF を使用したサンプル コードについては、『MSPM0 SDK サンプル ガイド』を参照してください。

6 まとめ

このアプリケーション ノートは、STM8 デバイスから MSPM0 に移行するエンジニアを支援することを目的としています。このドキュメントでは、エコシステム、CPU、アナログ／デジタル ペリフェラルの観点から、STM8 デバイスと MSPM0 の違いを比較しています。また、このアプリケーション ノートでは実例を交えて、MSPM0 をすぐに始めて開発の進行をスピードアップできるように、詳しく説明しています。

7 参考資料

- テキサス・インスツルメンツ、『MSPM0 SDK ユーザー ガイド』
- テキサス・インスツルメンツ、『MSPM0 ツール ガイド』
- テキサス・インスツルメンツ、『Driverlib API ガイド』
- テキサス・インスツルメンツ、『Code Composer Studio (CCS)』
- テキサス・インスツルメンツ、『MSPM0 L シリーズ 32MHz マイコン技術参照マニュアル』、技術参照マニュアル
- テキサス・インスツルメンツ、『MSPM0 C シリーズ 24MHz マイコン』、技術参照マニュアル
- IAR、IAR ウェブサイト
- Keil、Keil ウェブサイト
- テキサス・インスツルメンツ、『CCS クイック スタート ガイド』
- テキサス・インスツルメンツ、『CCS トレーニング動画』
- テキサス・インスツルメンツ、『CCS ユーザーズ ガイド』
- テキサス・インスツルメンツ、『IAR クイック スタート ガイド』
- テキサス・インスツルメンツ、『IAR トレーニング動画』
- IAR、IAR ユーザーズ ガイド
- Keil、Keil クイック スタート ガイド
- Keil、Keil トレーニング動画
- Keil、Keil を始める
- テキサス・インスツルメンツ、『SYSCONFIG IDE』
- テキサス・インスツルメンツ、『MSPM0 SysConfig ガイド』
- テキサス・インスツルメンツ、『XDS110 デバッグプローブ』
- テキサス・インスツルメンツ、『J-Link デバッグプローブのページ』
- テキサス・インスツルメンツ、『LP-MSPM0G3507 LaunchPad 開発キット』
- テキサス・インスツルメンツ、『LP-MSPM0L1306 LaunchPad 開発キット』
- テキサス・インスツルメンツ、『LP-MSPM0C1104 LaunchPad 開発キット』

8 改訂履歴

資料番号末尾の英字は改訂を表しています。その改訂履歴は英語版に準じています。

Changes from Revision * (December 2023) to Revision A (May 2025)	Page
• ドキュメント全体を通して STM8 デバイスとの比較対象を MSPM0H シリーズに更新	4
• ドキュメント全体を通して MSPM0C シリーズの仕様を更新	4
• IDE のバージョンや MSP フラッシュツールなど、新しいエコシステムを更新	5

重要なお知らせと免責事項

テキサス・インスツルメンツは、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、テキサス・インスツルメンツ製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した テキサス・インスツルメンツ製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとします。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている テキサス・インスツルメンツ製品を使用するアプリケーションの開発の目的でのみ、テキサス・インスツルメンツはその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。テキサス・インスツルメンツや第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、テキサス・インスツルメンツおよびその代理人を完全に補償するものとし、テキサス・インスツルメンツは一切の責任を拒否します。

テキサス・インスツルメンツの製品は、テキサス・インスツルメンツの販売条件、または ti.com やかかる テキサス・インスツルメンツ製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。テキサス・インスツルメンツがこれらのリソースを提供することは、適用されるテキサス・インスツルメンツの保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、テキサス・インスツルメンツはそれらに異議を唱え、拒否します。

郵送先住所:Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

重要なお知らせと免責事項

テキサス・インスツルメンツは、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、 テキサス・インスツルメンツ製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した テキサス・インスツルメンツ製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとします。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている テキサス・インスツルメンツ製品を使用するアプリケーションの開発の目的でのみ、 テキサス・インスツルメンツはその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。 テキサス・インスツルメンツや第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、 テキサス・インスツルメンツおよびその代理人を完全に補償するものとし、 テキサス・インスツルメンツは一切の責任を拒否します。

テキサス・インスツルメンツの製品は、 テキサス・インスツルメンツの販売条件、または ti.com やかかる テキサス・インスツルメンツ製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。 テキサス・インスツルメンツがこれらのリソースを提供することは、適用される テキサス・インスツルメンツの保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、 テキサス・インスツルメンツはそれらに異議を唱え、拒否します。

郵送先住所：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265